目　　录

第一篇　统计工具箱

第四篇　样条工具箱

第五篇　信号处理工具箱

第六篇 曲线拟合工具箱

第一篇　统计工具箱

第 1 章　统计工具箱简介

自然界和人类社会中会发生各种各样的现象，其中有的现象在一定条件下是必然要发生的，有的则表现出一定的随机性，但总体上又有一定的规律可循。一般称前者为确定性事件，后者为不确定性事件（或称随机事件）。概率论和数理统计就是研究和揭示不确定事件统计规律性的一门数学学科。

作为一个实用性很强的数学分支，概率论和数理统计的理论和方法已经广泛应用于管理、经济、心理、教育、体育、医学、生物、化学、机械、水文、地质、林业、气象、工业生产、建筑、通信、自动控制等几乎所有的社会和技术科学领域。

MATLAB 6.5 的统计工具箱相对于前面一些版本，改进较大。目前已经可以与 SPSS，SAS 等软件的统计功能相媲美。该工具箱是一个建立在 MATLAB 数值计算环境上的工具集，它支持随机数生成、曲线拟合、试验设计和统计过程控制等很多常见的统计任务。它提供了两类工具：概率统计函数和图形交互工具。

第 1 类工具由很多函数组成，这些函数可以从命令行或自己的应用程序中调用。它们大部分是 MATLAB M 文件，实现了一定的统计算法。假设函数名称为 function_name，在把该函数所在的路径设置为当前路径以后，在命令行中输入下面的语句，可以查看该函数的代码。

```
type function_name
```

将工具箱中的任何函数复制到其他路径并重命名以后修改它，可以改变它的行为方式。还可以在工具箱中添加自己的 M 文件来扩展工具箱。

第 2 类工具是图形交互工具。利用它们，可以以图形用户界面的形式实现很多函数的功能，方便而且直观。

1.1　统计工具箱的内容

MATLAB6.5 提供的工具箱内容非常丰富，涉及了当前大学生和研究生概率统计教材中的所有课题，新版本甚至增加了 K 均值聚类和决策树等内容，可以为数据挖掘中的某些任务提供解决方案。具体地讲，统计工具箱中包括以下几方面的内容。

- 概率分布：给出了常见的 20 种概率分布类型的概率密度函数、累加分布函数（分布函数）、逆累加分布函数、参数估计函数、随机数生成函数和统计量计算函数。
- 参数估计：提供了多种分布类型分布参数及其置信区间的估计方法。
- 样本描述：提供了描述中心趋势和离散趋势的统计量函数，缺失数据条件下的样本描述方法以及其他一些统计量计算函数。

● 方差分析：包括单因子方差分析、双因子方差分析和多因子方差分析。

● 多元方差分析：包括单因素多元方差分析、分组聚类和多元比较等。

● 回归分析：包括多元线性回归（含逐步回归）、岭回归、一般线性模型拟合、多项式拟合、稳健回归、响应面分析（含二维响应面分析和多维响应面分析）、非线性回归。

● 假设检验：包括单样本 t 检验、双样本 t 检验和 z 检验。

● 分布的检验：包括 Jarque-Bera 正态性检验、Kolmogorov-smirnov 单样本检验、Kolmogorov-smirnov 双样本检验和 Lilliefors 正态性检验。

● 非参数检验：包括 friedman 检验、Kruskalwallis 检验、秩和检验、符号秩检验和符号检验。

● 判别分析。

● 聚类分析：包括系统聚类和 K 均值聚类两种。

● 主成分分析。

● 因子分析。

● 决策树。

● 统计过程控制：提供了常用的过程管理图和过程性能图。

● 试验设计：包括完全析因设计、不完全析因设计、响应面设计和 D-优化设计。

● 统计图：包括箱形图、经验累加分布函数图、误差条图、函数交互等值线图、交互画线、交互点标注、散点矩阵图、散点图、添加最小二乘拟合线、正态概率图、帕累托图、q-q 图、回归个案次序图、参考多项式曲线、添加参考线、交互插值等值线图和威布尔图。

1.2 数学符号约定

在后续章节中，讲解统计工具箱中的函数时难免要提到各种数学概念和公式。表 1-1 列出了经常用到的数学符号的说明。

表 1-1 统计工具箱中的数学符号约定

数 学 符 号	说 明
β	线性模型中的参数
$E(x)$	x 的期望值
$f(x\|a,b)$	概率密度函数。x 是独立变量，a 和 b 是固定参数
$F(x\|a,b)$	累积分布函数
$I([a, b])$或 $I[a, b]$	指示函数
p 和 q	p 是某些事件发生的概率，q 是事件不发生的概率，q=1−p

第 2 章　概 率 分 布

试验得到的数据通常呈现一定的规律性，引入随机变量以后，可以将随机数据表达为随机变量的函数。常见的随机变量有离散型随机变量和连续型随机变量两种。

① 当变量全部可以取到的值是有限个或可列无限多个时，称为离散型随机变量。

② 如果对于随机变量 X 的分布函数 $F(x)$，存在非负函数 $f(x)$，使得对于任意实数 x 有

$$F(x)=\int_{-\infty}^{x} f(t)\,\mathrm{d}t \tag{2-1}$$

则称 X 为连续型随机变量。

对应于离散型随机变量和连续型随机变量，有离散型概率分布函数和连续型概率分布函数。常见的概率分布函数如表 2-1 所示。

表 2-1　常见概率分布函数

离散型概率分布	连续型概率分布
二项分布	正态分布
负二项分布	Γ（伽马）分布
几何分布	指数分布
超几何分布	χ^2（卡方）分布
泊松分布	威布尔分布
离散均匀分布	瑞利分布
	β（贝塔）分布
	对数正态分布
	柯西分布
	t 分布
	F 分布
	连续均匀分布

2.1　概率密度函数

2.1.1　基本数学原理

对于离散型概率分布和连续型概率分布，二者的概率密度函数定义有所不同。式（12.1）中，函数 $f(x)$ 称为 X 的概率密度函数。该函数具有以下性质：

① $f(x)\geqslant 0$。

② $\int_{-\infty}^{\infty} f(x)\mathrm{d}x=1$。

③ $P\{x_1\leqslant X\leqslant x_2\}=F(x_2)-F(x_1)=\int_{x_1}^{x_2} f(x)\mathrm{d}x,(x_1\leqslant x_2)$。

④ 若 $f(x)$ 在点 x 处连续，则有 $F'(x)=f(x)$。

对于离散型概率分布，则不称其为概率密度函数，而叫做概率分布或分布律。设离散型随机变量 X 所有可能取的值为 $x_k(k=1,2,\cdots)$，x 取各个可能值的概率，即事件$\{X=x_k\}$的概率为

$$P\{X=x_k\}=p_k, \quad k=1,2,\cdots$$

p_k 称为分布律。它有下面两个性质：

① $p_k \geqslant 0, k=1,2,\ldots,$

② $\sum_{k=1}^{\infty} p_k = 1$ 。

2.1.2 有关函数介绍

对于连续型概率分布函数，表 2-2 给出了对应函数的概率密度函数及其数学意义和调用格式。下面选择正态分布概率密度函数和指数分布概率密度函数进行重点介绍。

表 2-2 常见分布的概率密度

函 数 名	对应的分布	数 学 意 义	调 用 格 式
betapdf	β 分布	$y=f(x\mid a,b)=\frac{1}{B(a,b)}x^{a-1}(1-x)^{b-1}I_{(0,1)}(x)\ (0<x<1)$	Y = betapdf(X,A,B)
binopdf	二项分布	$y=f(x\mid n,p)=\binom{n}{x}p^x q^{(1-x)}I_{(0,1,\ldots n)}(x)$	Y=binopdf(X,N,P)
chi2pdf	卡方分布	$y=f(x\mid v)=\frac{x^{(v-2)/2}\mathrm{e}^{-x/2}}{2^{v/2}\Gamma(v/2)}$	Y=chi2pdf(X,V)
exppdf	指数分布	$y=f(x\mid\mu)=\frac{1}{\mu}e^{-\frac{x}{\mu}}$	Y=exppdf(X,MU)
fpdf	F 分布	$y=f(x\mid v_1,v_2)=\frac{\Gamma\left[\frac{v_1+v_2}{2}\right]}{\Gamma\left(\frac{v_1}{2}\right)\Gamma\left(\frac{v_2}{2}\right)}\left(\frac{v_1}{v_2}\right)^{\frac{v_1}{2}}\frac{x^{\frac{v_1-2}{2}}}{\left[1+\left(\frac{v_1}{v_2}\right)x\right]^{\frac{v_1+v_2}{2}}}$	Y=fpdf(X,V1,V2)
gampdf	伽马分布	$y=f(x\mid a,b)=\frac{1}{b^a\Gamma(a)}x^{a-1}\mathrm{e}^{-\frac{x}{b}}$	Y=gampdf(X,A,B)
geopdf	几何分布	$y=f(x\mid p)=pq^xI_{(0,1,K)}(x)$ 其中，$q=1-p$	Y=geopdf(X,P)
hygepdf	超几何分布	$y=f(x\mid M,K,n)=\frac{\binom{K}{x}\binom{M-K}{n-x}}{\binom{M}{n}}$	Y=hygepdf(X,M,K,N)
normpdf	正态（高斯）分布	$y=f(x\mid\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}\mathrm{e}^{\frac{-(x-\mu)^2}{2\sigma^2}}$	Y=normpdf(X,MU, SIGMA)
lognpdf	对数正态分布	$y=f(x\mid\mu,\sigma)=\frac{1}{x\sigma\sqrt{2\pi}}\mathrm{e}^{\frac{-(\ln x-\mu)^2}{2\sigma^2}}$	Y=lognpdf(X,MU, SIGMA)
nbinpdf	负二项分布	$y=f(x\mid r,p)=\binom{r+x+1}{x}p^xq^xI_{(0,1,\ldots)}(x)$ 式中，$q=1-p$	Y=nbinpdf(X,R,P)
ncfpdf	非中心 F 分布	假设随机变量 $\chi_1^2(\mu)$ 服从自由度为 f_1、非中心参数为 μ 的非中心 χ^2 分布，χ_2^2 服从自由度为 f_2 的 χ^2 分布，且 $\chi_1^2(\mu)$ 和 χ_2^2 相互独立，则随机变量 $F=\frac{\chi_1^2(\mu)/f_1}{\chi_2^2/f_2}$ 的分布称为自由度为（f_1，f_2）、非中心参数为 μ 的非中心 F 分布	Y=ncfpdf(X,NU1,NU2, DELTA)

续表

函 数 名	对应的分布	数 学 意 义	调 用 格 式
nctpdf	非中心 t 分布	如果 U 服从参数为 μ 和 1 的正态分布，$\chi^2_{(v)}$ 服从自由度为 v 的 χ^2 分布，并且 U 与 $\chi^2_{(v)}$ 相互独立，则称随机变量 $t(\mu)=\frac{U+\mu}{\sqrt{\chi^2_{(v)}/v}}$ 的分布为自由度为 v、非中心参数为 μ 的非中心 t 分布	Y=nctpdf(X,V,DELTA)
ncx2pdf	非中心 χ^2 分布	如果随机变量 X_i 服从参数为 μ_i（i=1,…,v）和 σ^2 的正态分布，并且相互独立，则随机变量 $\chi^2(\mu)=(X_1^2+\cdots+X_v^2)/\sigma^2$ 所服从的分布称为自由度为 v、非中心参数为 $\mu^2=(\mu_1^2+\cdots+\mu_v^2)/\sigma^2$ 的非中心 χ^2 分布	Y=ncx2pdf(X,V,DELTA)
poisspdf	泊松分布	$y=f(x\|\lambda)=\frac{\lambda^x}{x!}\mathrm{e}^{-\lambda}I_{(0,1,K)}(x)$	Y=poisspdf(X,LAMBDA)
raylpdf	瑞利分布	$y=f(x\|b)=\frac{x}{b^2}\mathrm{e}^{\left(\frac{-x^2}{2b^2}\right)}$	Y=raylpdf(X,B)
tpdf	学生氏 t 分布	$y=f(x\|v)=\frac{\Gamma\left(\frac{v+1}{2}\right)}{\Gamma\left(\frac{v}{2}\right)}\frac{1}{\sqrt{v\pi}}\frac{1}{\left(1+\frac{x^2}{v}\right)^{\frac{v+1}{2}}}$	Y=tpdf(X,V)
unidpdf	离散均匀分布	$y=f(x\|N)=\frac{1}{N}I_{(1,\ldots,N)}(x)$	Y=unidpdf(X,N)
unifpdf	连续均匀分布	$y=f(x\|a,b)=\frac{1}{b-a}I_{[a,b]}(x)$	Y=unifpdf(X,A,B)
weibpdf	威布尔分布	$y=f(x\|a,b)=abx^{b-1}\mathrm{e}^{-ax^b}I_{(0,\infty)}(x)$	Y=weibpdf(X,A,B)

1．正态概率密度函数

用 normpdf 函数计算正态概率密度函数。该函数的调用格式为：Y=normpdf (X, MU, SIGMA)，计算参数为 MU 和 SIGMA 的数据 X 的正态概率密度函数。参数 SIGMA 必须为正。

正态概率密度函数的计算公式为

$$y=f(x\mid\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}\mathrm{e}^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

似然函数为概率密度函数，它被视为参数的函数。参数的最大似然估计量（MLE）是使 x 处的似然函数最大化的值。

若 x 服从标准正态分布，则 $x+\mu$ 也服从均值为 μ，标准差为 σ 的正态分布。相反地，若 y 服从参数为 μ 和 σ 的正态分布，则 $x=(y-\mu)/\sigma$ 服从标准正态分布。

【例 2-1】

```
mu=[0:0.1:2];
[y i]=max(normpdf(1.5,mu,1));
MLE=mu(i)
MLE=
     1.5000
```

2．指数概率密度函数

用 exppdf 函数计算指数概率密度函数。语法格式为：Y=exppdf (X, MU)，计算 x 处的参数为 MU 的指数概率密度函数。MU 参数必须为正。指数概率密度函数为

$$y = f(x\,|\,\mu) = \frac{1}{\mu}\mathrm{e}^{-\frac{x}{\mu}}$$

指数概率密度函数等价于第 1 个参数（a ）等于 1 时的伽马概率密度函数。

【例 2-2】

```
y = exppdf(5,1:5)
y =
   0.0067   0.0410   0.0630   0.0716   0.0736
y = exppdf(1:5,1:5)
y =
   0.3679   0.1839   0.1226   0.0920   0.0736
```

2.2　累加分布函数

2.2.1　基本数学原理

对于连续型随机变量，其分布函数的定义为：若 X 为随机变量，x 为任意实数，则函数

$$F(x) = p\{X \leqslant x\}$$

被称为 X 的分布函数。如果知道 X 的分布函数，就知道 X 落在任一区间（$x_1\ x_2$］上的概率。

分布函数 $F(x)$具有以下一些性质：

① $F(x)$是不减函数；

② $0 \leqslant F(x) \leqslant 1$，且

$$F(-\infty) = \lim_{x\to -\infty} F(x) = 0,$$

$$F(\infty) = \lim_{x\to \infty} F(x) = 1;$$

③ $F(x+0) = F(x)$，即 $F(x)$是右连续的。

2.2.2　有关函数介绍

1．累加正态分布函数

用 normcdf 函数计算累加正态分布函数。调用格式为：P = normcdf (X, MU, SIGMA)，计算服从参数为 MU 和 SIGMA 的正态分布数据 X 的累加分布函数。参数 SIGMA 必须为正。

累加正态分布函数为：

$$p = F(x\,|\,\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}\int_{-\infty}^{x}\mathrm{e}^{\frac{-(t-\mu)^2}{2\sigma^2}}\mathrm{d}t$$

结果 p 为取自参数为μ和σ的正态分布的单个观测量落在区间(-∞ x)中的概率。

【例 2-3】　下面的例子求取自标准正态分布的一个观测量落在区间[-1 1]中的概率。

```
p = normcdf([-1 1]);
p(2) - p(1)
ans =
     0.6827
```

更一般地，若观测量取自参数为σ和μ的正态分布，则它落在该区间中的概率为 68%。

2．累加指数分布函数

用 expcdf 函数计算累加指数分布函数。调用格式为：P = expcdf (X, MU)，计算参数为

MU 的数据 X 的累加指数分布函数。指数 MU 必须为正。

累加指数分布函数的计算公式为

$$p=F(x\,|\,\mu)=\int_0^x \frac{1}{\mu}\mathrm{e}^{-\frac{t}{\mu}}\mathrm{d}t=1-\mathrm{e}^{-\frac{x}{\mu}}$$

结果 p 为取自指数分布样本的单个观测量落在区间[0 x]中的概率。

【例 2-4】 指数为 μ 的指数分布数据的中值等于 μ*log(2)，下例进行演示。

```
mu = 10:10:60;
p = expcdf(log(2)*mu,mu)
p =
    0.5000    0.5000    0.5000    0.5000    0.5000    0.5000
```

下面计算指数分布随机变量小于或等于均值 μ 的概率。

```
mu = 1:6;
x = mu;
p = expcdf(x,mu)
p =
    0.6321    0.6321    0.6321    0.6321    0.6321    0.6321
```

表 2-3 中为常见分布的累加函数及其调用格式。

表 2-3　常见分布的累加函数

函数名	累加函数对应的分布	数 学 意 义	调 用 格 式
betacdf	β 分布	$p=F(x\|a,b)=\dfrac{1}{B(a,b)\int_0^x t^{a-1}(1-t)^{b-1}\mathrm{d}t}$	P=betacdf(X, A, B)
binocdf	二项分布	$y=F(x\|n,p)=\sum_{i=0}^{x}\binom{n}{i}p^i q^{(1-i)}I_{(0,1,\cdots,n)}(i)$	Y=binocdf(X, N, P)
chi2cdf	卡方分布	$p=F(x\|v)=\int_0^x \dfrac{t^{(v-2)/2}\mathrm{e}^{-t/2}}{2^{v/2}\Gamma(v/2)}\mathrm{d}t$	P=chi2cdf(X, V)
expcdf	指数分布	$p=F(x\|\mu)=\int_0^x \dfrac{1}{\mu}\mathrm{e}^{-\frac{x}{\mu}}\mathrm{d}t=1-\mathrm{e}^{-\frac{x}{\mu}}$	P=expcdf(X, MU)
fcdf	F 分布	$F(x\|v_1,v_2)=\int \dfrac{\Gamma\left[\frac{v_1+v_2}{2}\right]}{\Gamma\left(\frac{v_1}{2}\right)\Gamma\left(\frac{v_2}{2}\right)}\left(\dfrac{v_1}{v_2}\right)^{\frac{v_1}{2}}\dfrac{t^{\frac{v_1-2}{2}}}{\left[1+\left(\frac{v_1}{v_2}\right)\right]^{-\frac{v_1+v_2}{2}}}\mathrm{d}t$	P=fcdf(X, V1, V2)
gamcdf	伽马分布	$p=F(x\|a,b)=\dfrac{1}{b^a\Gamma(a)}\int_0^x t^{a-1}\mathrm{e}^{-\frac{1}{b}}\mathrm{d}t$	P=gamcdf(X,A,B)
geocdf	几何分布	$y=F(x\|p)=\sum_{i=0}^{floor(x)}pq^i, q=1-p$	Y=geocdf(X,P)
hygecdf	超几何分布	$p=F(x\|M,K,N)=\sum_{i=0}^{x}\dfrac{\binom{K}{i}\binom{M-K}{N-i}}{\binom{M}{N}}$	P=hygecdf(X,M,K,N)
logncdf	对数正态分布	$p=f(x\|\mu,\sigma)=\dfrac{1}{x\sigma\sqrt{2\pi}}\int_0^x \dfrac{\mathrm{e}^{\frac{-(\ln t-\mu)^2}{2\sigma^2}}}{t}\mathrm{d}t$	P=logncdf(X,MU,SIGMA)

续表

函数名	累加函数对应的分布	数学意义	调用格式
nbincdf	负二项分布	$y=F(x\mid r,p)=\sum_{i=0}^{x}\begin{pmatrix}r+i+1\\i\end{pmatrix}p^r q^i I_{(0,1,\cdots)}(i)$	Y=nbincdf(X,R,P)
ncfcdf	非中心 F 分布	$F(x\mid v_1,v_2,\delta)=\sum\left(\frac{\left(\frac{1}{2}\delta\right)^j}{j!}\mathrm{e}^{-\frac{\delta}{2}}\right)I\left(\frac{\left(\frac{v_1x}{v_2+v_1x}\right)}{v_1/2+j},v_2/2\right)$	P=ncfcdf(X,NU1,NU2, DELTA)
nctcdf	非中心 t 分布	$\Pr((-t)<x<t\mid v,\delta)=\sum\left(\frac{\left(\frac{1}{2}\delta\right)^j}{j!}\mathrm{e}^{\frac{\delta^2}{2}}\right)\left(\frac{\left(\frac{x^2}{v+x^2}\right)}{v/2+j},v/2\right)$	P=nctcdf(X,NU,DELTA)
ncx2cdf	非中心卡方分布	$F(x\mid v,\delta)=\sum\left(\frac{\left(\frac{1}{2}\delta\right)^j}{j!}\mathrm{e}^{\frac{\delta}{2}}\right)\Pr[\chi^2_{v+2j}\leqslant x]$	P=ncx2cdf(X,V,DELTA)
normcdf	正态(高斯)分布	$p=f(x\mid\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}\int_{-\infty}^{x}\mathrm{e}^{\frac{-(t-\mu)^2}{2\sigma^2}}\mathrm{d}t$	P=normcdf(X,MU,SIGMA)
poisscdf	泊松分布	$p=F(x\mid\lambda)=\mathrm{e}^{-\lambda}\sum_{i=0}^{floor(x)}\frac{\lambda^i}{i!}$	P=poisscdf(X,LAMBDA)
raylcdf	瑞利分布	$y=F(x\mid b)=\int_0^x\frac{1}{b^2}\mathrm{e}^{\left(\frac{-t^2}{2b^2}\right)}\mathrm{d}t$	P=raylcdf(X,B)
tcdf	学生氏 t 分布	$p=F(x\mid v)=\int\frac{\Gamma\left(\frac{v+1}{2}\right)}{\Gamma\left(\frac{v}{2}\right)}\frac{1}{\sqrt{v\pi}}\frac{1}{\left(1+\frac{t^2}{v}\right)^{\frac{v+1}{2}}}\mathrm{d}t$	P=tcdf(X,V)
unidcdf	离散均匀分布	$p=F(x\mid N)=\frac{floor(x)}{N}I_{(1,\cdots,N)}(x)$	P=unidcdf(X,N)
unifcdf	连续均匀分布	$p=F(x\mid a,b)=\frac{x-a}{b-a}I_{[a,b]}(x)$	P=unifcdf(X,A,B)
weibcdf	威布尔分布	$p=F(x\mid a,b)=\int_0^x abt^{b-1}\mathrm{e}^{-at^b}\mathrm{d}t=1-\mathrm{e}^{-ax^b}I_{(0,\infty)}(x)$	P=weibcdf(X,A,B)

2.3　参数估计

2.3.1　基本数学原理

参数估计的内容包括点估计和区间估计。

点估计是用单个数值作为参数的估计，常用的方法有矩法和最大似然法。

① 矩法：某些情况下，待估参数往往是总体原点矩或原点矩的函数，此时可以用取自该总体的样本的原点矩或样本原点矩的函数值作为待估参数的估计，这种方法称为矩法。如，样本均值总是总体均值的矩估计量，样本方差总是总体方差的矩估计量，样本标准差总是总体标准差的矩估计量。

② 最大似然法：最大似然法是在待估参数的可能取值范围内进行挑选，使似然函数值

（即样本取固定观察值或样本取值落在固定观察值邻域内的概率）最大的那个参数值即为最大似然估计量。由于最大似然估计法得到的估计量通常不仅仅满足无偏性、有效性等基本条件，还能保证其为充分统计量，所以，在点估计和区间估计中，一般推荐使用最大似然法。

区间估计不仅仅给出参数的近似取值，还给出了该取值的误差范围。求参数的区间估计，首先要求出该参数的点估计，然后构造一个含有该参数的随机变量，并根据一定的置信水平求该估计值的误差范围。

2.3.2 有关函数介绍

1．正态分布数据的参数估计

用 normfit 函数对正态分布数据进行参数估计，求参数的置信区间。其调用格式和说明如下所示：

● [muhat, sigmahat, muci, sigmaci] = normfit(X) 对于给定的服从正态分布的数据矩阵 X，返回参数μ和σ的估计值 muhat 和 sigmahat。muci 和 sigmaci 为μ和σ的 95%置信区间。muci 和 sigmaci 向量分别有两行，其列数与数据矩阵 X 的列数相同。上下两行的数据分别为置信区间的下限和上限。

● [muhat, sigmahat, muci, sigmaci] = normfit(X, alpha) 进行参数估计并计算 100（1-alpha）置信区间。如 alpha = 0.01 时，给出 99%置信区间。

【例 2-5】 本例中，数据为两列随机正态矩阵。两列都有μ=10 和σ = 2。

```
r = normrnd(10,2,100,2);
[mu,sigma,muci,sigmaci] = normfit(r)
mu =
   10.1455   10.0527
sigma =
    1.9072    2.1256
muci =
    9.7652    9.6288
   10.5258   10.4766
sigmaci =
    1.6745    1.8663
    2.2155    2.4693
```

2．贝塔（β）分布数据的参数估计

用 betafit 函数对服从贝塔分布的数据进行参数估计，并计算置信区间。其语法格式有下面两种：

phat = betafit (x)

[phat,pci] = betafit (x, alpha)

其中，betafit 函数计算服从贝塔分布的数据 x 的参数的最大似然估计，有两个输出变量。该函数也可以以 2×2 矩阵的形式返回参数的置信区间，矩阵的第 1 列包括 A 参数的下界和上界，第 2 列包括 B 参数的下界和上界。

alpha 参数为输入变量的可选项，它控制置信区间的宽度。在默认情况下，alpha 等于 0.05，对应于 95%置信区间。

【例 2-6】 本例首先用 betarnd 函数生成 100 个服从贝塔分布的数据。假设参数真值分别为 4 和 3，现利用 betafit 函数进行参数估计，并计算参数的置信区间。

```
r = betarnd(4,3,100,1);
[p,ci] = betafit(r,0.01)
p =
    3.9010    2.6193
ci =
    2.5244    1.7488
    5.2777    3.4899
```

3．负贝塔对数似然函数

用 betalike 函数计算负贝塔对数似然函数。该函数的语法格式及说明如下所示：

● logL = betalike(params, data) 对于给定的数据 data，返回两个 beta 参数的贝塔对数似然负函数。logL 的长度为 data 的长度。

● h[logL, info] = betalike(params, data) 该形式还返回费歇尔信息矩阵 info。费歇尔信息矩阵的对角线上为各参数的渐进方差。

betalike 函数是一个工具函数，用于求取贝塔函数的最大似然估计。要求数据样本满足相互独立的似然假设。由于 betalike 函数返回负的伽马对数似然函数，所以使用 fmins 函数使 betalike 最小化的效果与使 likelihood 函数最大化的效果相同。

【例 2-7】 继续使用 betafit 函数的例子。

```
r = betarnd(4,3,100,1);
[logl,info] = betalike([3.9010 2.6193],r)
logl =
  -33.0514
info =
    0.2856    0.1528
    0.1528    0.1142
```

4．最大似然估计

用 mle 函数进行最大似然估计。该函数的语法格式和说明如下所示：

● phat = mle('dist', data) 用 data 向量中的样本返回‘dist’指定的分布的最大似然估计（MLE）。

● [phat, pci] = mle('dist', data) 返回最大似然估计和 95%置信区间。

● [phat, pci] = mle('dist', data, alpha) 返回指定分布的最大似然估计值和 100（1-alpha）置信区间。

● [phat, pci] = mle('dist', data, alpha, p1) 该形式仅用于二项分布，其中 p1 为试验次数。

【例 2-8】

```
rv = binornd(20,0.75)
rv =
    16
[p,pci] = mle('binomial',rv,0.05,20)
p =
    0.8000
pci =
    0.5634
    0.9427
```

表 2-4 中为常见分布的参数估计函数及其调用格式。

表 2-4　常见分布的参数估计函数及其调用格式

函 数 名	参数估计对应的分布	调 用 格 式
betafit	贝塔分布	phat = betafit(x) [phat, pci] = betafit(x, alpha)
betalike	贝塔对数似然函数	logL=betalike(params, data) [logL, info]=betalike(params, data)
binofit	二项分布	phat=binofit(x, n) [phat, pci]=binofit(x, n) [phat, pci]=binofit(x, n, alpha)
expfit	指数分布	muhat=expfit(x) [muhat, muci]=expfit(x) [muhat, muci]=expfit(x, alpha)
gamfit	伽马分布	phat=gamfit(x) [phat, pci]=gmfit(x) [phat, pci]=gamfit(x, alpha)
gamlike	伽马似然函数	logL=gamlike(params, data) [logL, info]=gamlike(params, data)
mle	最大似然估计	phat=mle('dist', data) [phat, pci]=mle('dist', data) [phat, pci]=mle('dist', data, alpha) [phat, pci]=mle('dist',data, alpha, p1)
normlike	正态对数似然函数	L=normlike(params,data)
normfit	正态分布	[muhat,sigmahat,muci,sigmaci]=normfit(X) [muhat,sigmahat,muci,sigmaci]=normfit(X,alpha)
poissfit	泊松分布	lambdahat=poissfit(X) [lambdahat,lambdaci]=poissfit(X) [lambdahat,lambdaci]=poissfit(X,alpha)
unifit	均匀分布	[ahat,bhat]=unifit(X) [ahat,bhat,ACI,BCI]=unifit(X) [ahat,bhat,ACI,BCI]=unifit(X,alpha)
weibfit	威布尔分布	phat=weibfit(x) [phat,pci]=weibfit(x) [phat,pci]=weibfit(x,alpha)
weiblike	威布尔对数似然函数	logL=weiblike(params,data) [logL,info]=weiblike(params,data)

2.4　逆累加分布函数

2.4.1　基本数学原理

逆累加分布函数是累加分布函数的逆函数。利用逆累加分布函数，可以求得满足给定概率时随机变量对应的置信区间的最小值和最大值。

2.4.2　有关函数介绍

1．累加正态分布函数的逆函数

用 norminv 函数计算累加正态分布函数的逆函数。

● X = norminv(P, MU, SIGMA) 计算 P 处参数为 MU 和 SIGMA 的累加正态函数的逆函数。SIGMA 必须为正，P 值必须属于[0 1]区间。

下面用累加正态分布函数的形式来定义正态逆函数。

$$x = F^{-1}(p \mid \mu, \sigma) = \{x : F(x \mid \mu, \sigma) = p\}$$

式中

$$p = F(x \mid \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} \mathrm{e}^{\frac{-(t-\mu)^2}{2\sigma^2}} \mathrm{d}t$$

结果 x 为上面参数为μ和σ的整型方程的解，其中，概率 p 是预先给定的。

【例 2-9】 求包含标准正态分布数据 95%的值的区间。

```
x = norminv([0.025 0.975],0,1)
x =
   -1.9600    1.9600
hxl = norminv([0.01 0.96],0,1)
xl =
   -2.3263    1.7507
```

区间 *x*l 也包含 95%的值，但它比 x 长。

2. 累加指数分布函数的逆函数

用 expinv 函数计算累加指数分布函数的逆函数。

● X = expinv(P,MU) 计算 P 处参数为 MU 的累加指数分布函数的逆函数。MU 必须为正，P 值必须界于 0 和 1 之间。

累加指数分布函数的逆函数为

$$x = F(p \mid \mu) = -\mu \ln(1-p)$$

结果 x 表示取自参数为μ的指数分布的观测值落在[0 x]区间内的概率为 p 时对应的值。

【例 2-10】 假设电灯泡的使用寿命服从 mu 等于 700 小时的指数分布，求电灯泡使用寿命的中值。

```
expinv(0.50,700)
ans =
    485.2030
```

所以，假设你买了一箱标明使用寿命为“700 小时”的电灯泡，若 700 小时为电灯泡的平均寿命，则其中一半的使用寿命不会超过 500 小时。

表 2-5 为常见分布的累加分布逆函数及其调用格式。

表 2-5 常见分布的累加分布逆函数

函数名	累加分布函数逆函数对应的分布	调 用 格 式
betainv	贝塔分布	X=betainv(P, A, B)
binoinv	二项分布	X=binoinv(Y, N, P)
chi2inv	卡方分布	X=chi2inv(P, V)
expinv	指数分布	X=expinv(P, MU)
finv	F 分布	X=finv(P, V1, V2)
gaminv	伽马分布	X=gaminv(P, A, B)
geoinv	几何分布	X=geoinv(Y, P)

续表

函数名	累加分布函数逆函数对应的分布	调 用 格 式
hygeinv	超几何分布	X=hygeinv(P, M, K, N)
logninv	对数正态分布	X=logncdf(X, MU, SIGMA)
nbininv	负二项分布	Y=nbincdf(X, R, P)
ncfinv	非中心 F 分布	P=ncfcdf(X, NU1, NU2, DELTA)
nctinv	非中心 t 分布	P=nctcdf(X, NU, DELTA)
ncx2inv	非中心卡方分布	P=ncx2cdf(X, V, DELTA)
icdf		
norminv	正态（高斯）分布	X=norminv(P, MU, SIGMA)
poissinv	泊松分布	X=poissinv(P, LAMBDA)
raylinv	瑞利分布	X=raylinv(P,B)
tinv	学生氏 t 分布	X=tinv(P,V)
unidinv	离散均匀分布	X=unidinv(P,N)
unifinv	连续均匀分布	X=unifcdf(X,A,B)
weibinv	威布尔分布	X=weibcdf(X,A,B)

2.5　随机数的生成

2.5.1　随机数生成的基本原理

生成给定分布的随机数，需要首先生成服从均匀分布的随机数。常用的生成均匀分布随机数的方法是同余法，其递推公式为

$$x_i = (ax_{i-1} + c) \bmod m$$

给定初值 x_0 后，可以迭代出均匀随机数 $x_1, x_2, ..., x_n$。对它们进行标准化（使随机数介于 0 和 1 之间）或极性标准化（使随机数介于-1 和 1 之间），即可得到均匀分布的随机数。

获得均匀分布的随机数以后，可以用多种方法构造基于该随机数的随机变量。常用的方法是反函数法，即利用随机变量 x 的分布函数 $F(x)$的反函数 $F^{-1}(x)$来推求随机变量。基本算法是：

- 产生均匀分布随机数 r；
- 令 $x= F^{-1}(x)$，然后返回。

下面结合正态分布随机变量的生成进行具体介绍。正态分布密度函数为

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right] \qquad (-\infty < x < +\infty)$$

式中，μ 为期望值,；σ^2 为方差。分布函数为

$$F(x) = \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{x} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right] \mathrm{d}x$$

$F(x)$为非可积函数，由中心极限定理，有

$$\frac{x-\mu}{\sigma} = \sqrt{\frac{12}{n}\left(\sum_{i=1}^{n} r_i - \frac{n}{2}\right)}$$

当 n=12 时，可达到较好精度，故

$$x=\left(\sum_{i=1}^{12}r_i-6\right)\sigma+\mu$$

该 $\boldsymbol{x}$ 就是基于均匀分布随机数 r_i 的服从正态分布的随机数。

2.5.2　有关函数介绍

1．正态分布的随机数

用 normrnd 函数生成服从正态分布的随机数。该函数的调用格式和说明为：

- R = normrnd(MU, SIGMA) 生成均值为 MU，标准差为 SIGMA 的正态分布随机数。
- R = normrnd(MU, SIGMA, m) 生成均值为 MU，标准差为 SIGMA 的正态分布随机数。m 为 1×2 的向量，包含 R 的行和列的维数。
- R = normrnd(MU, SIGMA, m, n) 生成均值为 MU，标准差为 SIGMA 的正态分布随机数。m 和 n 分别为行和列的维数。

【例 2-11】

```
n1 = normrnd(1:6,1./(1:6))
n1 =
    2.1650    2.3134    3.0250    4.0879    4.8607    6.2827
n2 = normrnd(0,1,[1 5])
n2 =
    0.0591    1.7971    0.2641    0.8717   -1.4462
n3 = normrnd([1 2 3;4 5 6],0.1,2,3)
n3 =
    0.9299    1.9361    2.9640
    4.1246    5.0577    5.9864
```

2．指定分布的随机数

用 random 函数生成指定分布的随机数。它是一个工具函数，通过将分布名称指定为参数，利用它可以生成服从各种分布的随机数。调用格式为：

- y = random('name', A1, A2, A3, m, n) 返回一个随机数矩阵。

'name'为一字符串，包含分布名；A1，A2 和 A3 为分布参数矩阵。有的分布不需要这么多参数。最后两个参数分别表示 x 矩阵和 y 矩阵的大小。如果分布参数为矩阵，则这些参数是可选的，但它们必须与其他矩阵变量的大小相匹配。

【例 2-12】

```
rn = random('Normal',0,1,2,4)
rn =
    1.1650    0.0751   -0.6965    0.0591
    0.6268    0.3516    1.6961    1.7971
rp = random('Poisson',1:6,1,6)
rp =
     0     0     1     2     5     7
```

表 2-6 中为常见分布随机数生成函数的调用格式。

表 2-6　常见分布随机数生成函数的调用格式

函　数	分　布	调　用　格　式		
		格 式 一	格 式 二	格 式 三
betarnd	贝塔分布	R=betarnd(A,B)	R=betarnd(A,B,m)	R=betarnd(A,B,m,n)
binornd	二项分布	R=binornd(N,P)	R=binornd(N,P,mm)	R=binornd(N,P,mm,nn)
chi2rnd	卡方分布	R=chi2rnd(V)	R=chi2rnd(V,m)	R=chi2rnd(V,m,n)
exprnd	指数分布	R=exprnd(MU)	R=exprnd(MU,m)	R=exprnd(MU,m,n)
frnd	F 分布	R=frnd(V1,V2)	R=frnd(V1,V2,m)	R=frnd(V1,V2,m,n)
gamrnd	伽马分布	R=gamrnd(A,B)	R=gamrnd(A,B,m)	R=gamrnd(A,B,m,n)
geornd	几何分布	R=geornd(P)	R=geornd(P,m)	R=geornd(P,m,n)
hygernd	超几何分布	R=hygernd(M,K,N)	R=hygernd(M,K,N,mm)	R=hygernd(M,K,N,mm,nn)
lognrnd	对数正态分布	R=lognrnd(MU,SIGMA)	R=lognrnd(MU,SIGMA,m)	R=lognrnd(MU,SIGMA,m,n)
nbinrnd	负二项分布	R=nbinrnd(R,P)	R=nbinrnd(R,P,m)	R=nbinrnd(R,P,m,n)
ncfrnd	非中心 F 分布	R=ncfrnd(NU1,NU2,DELTA)	R=ncfrnd(NU1,NU2,DELTA,m)	R=ncfrnd(NU1,NU2,DELTA,m,n)
nctrnd	非中心 t 分布	R=nctrnd(V,DELTA)	R=nctrnd(V,DELTA,m)	R=nctrnd(V,DELTA,m,n)
ncx2rnd	非中心卡方分布	R=ncx2rnd(V,DELTA)	R=ncx2rnd(V,DELTA,m)	R=ncx2rnd(V,DELTA,m,n)
normrnd	正态（高斯）分布	R=normrnd(MU,SIGMA)	R=normrnd(MU,SIGMA,m)	R=normrnd(MU,SIGMA,m,n)
poissrnd	泊松分布	R=poissrnd(LAMBDA)	R=poissrnd(LAMBDA,m)	R=poissrnd(LAMBDA,m,n)
raylrnd	瑞利分布	R=raylrnd(B)	R=raylrnd(B,m)	R=raylrnd(B,m,n)
trnd	学生氏 t 分布	R=trnd(V)	R=trnd(V,m)	R=trnd(V,m,n)
unidrnd	离散均匀分布	R=unidrnd(N)	R=unidrnd(N,mm)	R=unidrnd(N,mm,nn)
unifrnd	连续均匀分布	R=unifrnd(N)	R=unifrnd(N,mm)	R=unifrnd(N,mm,nn)
weibrnd	威布尔分布	R=weibrnd(A,B)	R=weibrnd(A,B,m)	R=weibrnd(A,B,m,n)

2.6　分布函数的统计量估计

1. 正态分布函数的统计量估计

对于正态分布，有

$$y = f(x \mid \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \mathrm{e}^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

式中，μ 为均值，σ^2 为方差。

用 normstat 函数求正态分布的均值和方差。其语法格式为：

- [M, V] = normstat (MU, SIGMA)

【例 2-13】

```
n = 1:5;
[m,v] = normstat(n'*n,n'*n)
m =
    1     2     3     4     5
    2     4     6     8    10
    3     6     9    12    15
    4     8    12    16    20
```

```
     5    10    15    20    25
v =
     1     4     9    16    25
     4    16    36    64   100
     9    36    81   144   225
    16    64   144   256   400
    25   100   225   400   625
```

2．指数分布函数的统计量估计

对于指数分布函数，有

$$y = f(x \mid \mu) = \frac{1}{\mu}\mathrm{e}^{-\frac{x}{\mu}}$$

式中，μ为均值，μ^2为方差。

用 expstat 函数计算指数分布的均值和方差。其语法格式为：

- [M, V] = expstat(MU)

【例 2-14】

```
[m,v] = expstat([1 10 100 1000])
m =
          1         10        100        1000
v =
          1        100      10000     1000000
```

表 2-7 中为常见分布的统计量估计函数及其调用格式。

表 2-7　常见分布的统计量估计

函数名	分 布	参 数 意 义		调 用 格 式
		均　值	方　差	
betastat	贝塔分布	$\frac{a}{a+b}$	$\frac{ab}{(a+b+1)(a+b)^2}$	[M, V]=betastat(A, B)
binostat	二项分布	np	npq	[M, V]=binostat(N, P)
chi2stat	卡方分布	n	$2n$	[M, V]=chi2stat(NU)
expstat	指数分布	μ	μ^2	[M, V]=expstat(MU)
fstat	F 分布	$\frac{v_2}{v_2-2}, v_2>2$	$\frac{2v_2^2(v_1+v_2-2)}{v_1(v_2-2)^2(v_2-4)}, v_2>4$	[M, V]=fstat(V1, V2)
gamstat	伽马分布	ab	ab^2	[M, V]=gamstat(A, B)
geostat	几何分布	q/p	q/p^2	[M, V]=geostat(P)
hygestat	超几何分布	$N\frac{K}{M}$	$N\frac{K}{M}\frac{M-K}{M}\frac{M-N}{M-1}$	[M,V]=hygestat(M,K,N)
lognstat	对数正态分布	$\mathrm{e}^{\left(\mu+\frac{\sigma^2}{2}\right)}$	$\mathrm{e}^{(2\mu+2\sigma^2)}-\mathrm{e}^{(2\mu+\sigma^2)}$	[M,V]=lognstat(MU,SIGMA)
nbinstat	负二项分布	$\frac{rq}{p}$	$\frac{rq}{p^2}$	[M,V]=nbinstat(R,P)
ncfstat	非中心 F 分布	$\frac{v_2(\delta+v_1)}{v_1(v_2-2)}, v_2>2$	$2\left(\frac{v_2}{v_1}\right)^2\left[\frac{(\delta+v_1)^2+(2\delta+v_1)(v_2-2)}{(v_2-2)^2(v_2-4)}\right], v_2>4$	[M,V]=ncfstat(NU1,NU2,DELTA)
nctstat	非中心 t 分布	$\frac{\delta(v/2)^{1/2}\Gamma((v-1)/2)}{\Gamma(v/2)}$	$\frac{v}{(v-2)}(1+\delta^2)-\frac{v}{2}\delta^2\left[\frac{\Gamma((v-1)/2)}{\Gamma(v/2)}\right]^2$	[M,V]=nctstat(NU,DELTA)

续表

函数名	分 布	参 数 意 义		调 用 格 式
		均　值	方　差	
ncx2stat	非中心卡方分布	$v+\delta$	$2(v+2\delta)$	[M,V]=ncx2stat(NU,DELTA)
normstat	正态（高斯）分布	μ	σ^2	[M,V]=normstat(MU,SIGMA)
poisstat	泊松分布	λ	λ	[M,V]=poisstat(LAMBDA)
raylstat	瑞利分布	$b\left(\frac{\pi}{2}\right)^{\frac{1}{2}}$	$\frac{2-\pi}{2}b^2$	[M,V]=raylstat(B)
tstat	学生氏 t 分布	若 v >1，则均值为 0；若 v =1，则均值不存在	$\frac{v}{v-2}, v>2$	[M,V]=tstat(NU)
unidstat	离散均匀分布	$\frac{N+1}{2}$	$\frac{N^2-1}{12}$	[M,V]=unidstat(N)
unifstat	连续均匀分布	$(a+b)/2$	$(b-a)/12$	[M,V]=unidstat(A,B)
weibstat	威布尔分布	$a^{-\frac{1}{b}}\Gamma(1-b^{-1})$	$a^{-\frac{2}{b}}\left[\Gamma(1+2b^{-1})-\Gamma^2(1+b^{-1})\right]$	[M,V]=weibstat(A,B)

第 3 章　样 本 描 述

3.1　概述

采集到大量的样本数据以后，常常需要用一些统计量来描述数据的集中程度和离散程度，并通过这些指标来对数据的总体特征进行归纳。

描述样本数据集中趋势的统计量有算术平均值、中位数、众数、几何均值、调和均值和截尾均值等。

描述样本数据离散趋势的统计量包括极差、平均差、平均绝对差、方差和标准差等。

此外还有峰度、偏度、分位数和相关系数等统计量也能描述样本数据的某些特征。

3.2　描述集中趋势的统计量

3.2.1　几何均值

样本数据 $x_1, x_2, ..., x_n$ 的几何均值 m 可以根据下式求得：

$$m = \left[\prod_{i=1}^{n} x_i\right]^{\frac{1}{n}}$$

用 geomean 函数计算样本的几何均值。

● m = geomean 函数计算样本的几何均值。对于向量而言，geomean(X)为数据 ***X*** 中元素的几何均值。对于矩阵而言，geomean(X)为一行向量，其中包含每一列数值的几何均值。

【例 3-1】　样本均值大于或等于样本的几何均值。

```
x = exprnd(1,10,6);
geometric = geomean(x)
geometric =
    0.7466    0.6061    0.6038    0.2569    0.7539    0.3478
average = mean(x)
average =
    1.3509    1.1583    0.9741    0.5319    1.0088    0.8122
```

3.2.2　调和均值

样本数据 $x_1, x_2, ..., x_n$ 的调和平均值 m 定义为

$$m = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}$$

用 harmmean 函数计算样本数据的调和平均值。

● m = harmmean 函数计算样本的调和平均值。对于向量而言，harmmean(X)函数为 ***X*** 中元素的调和平均值。对于矩阵而言，harmmean(X)函数为一包含每列元素调和平均值的行向量。

【例 3-2】 样本均值大于或等于样本的调和平均值。

```
x = exprnd(1,10,6);
harmonic = harmmean(x)
harmonic =
    0.3382    0.3200    0.3710    0.0540    0.4936    0.0907
average = mean(x)
average =
    1.3509    1.1583    0.9741    0.5319    1.0088    0.8122
```

3.2.3 算术平均值

样本数据 $x_1, x_2, ..., x_n$ 的算术平均值可用下式定义：

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

用 mean 函数计算向量和矩阵中元素的均值。语法格式为：

- m = mean(X)

对于向量，mean(x)为 ***X*** 中元素的均值。对于矩阵，mean(X)为包含 ***X*** 中每列元素均值的行向量。

【例 3-3】 下面的命令行生成 5 个包含 100 个服从标准正态分布的随机数的样本。

```
x = normrnd(0,1,100,5);
xbar = mean(x)
xbar =
   0.0727    0.0264    0.0351    0.0424    0.0752
```

3.2.4 中值

用 median 函数计算向量和矩阵中元素的中值。

- m = median(X) 计算中值，即样本的 50%位置处的值。中值是样本数据中心趋势的稳健估计，因为异常值的影响较小。

对于向量，median(X)为向量 ***X*** 中元素的中值。对于矩阵，median(X)为包含每一列中元素中值的行向量。计算中值需要首先进行排序，因此计算大型矩阵的中值向量时将比较费时。

【例 3-4】

```
xodd = 1:5;
modd = median(xodd)
modd =
    3
meven = median(xeven)
meven =
    2.5000
```

下例演示中值对于异常值的稳健性。

```
xoutlier = [x 10000];
moutlier = median(xoutlier)
moutlier =
    3
```

3.2.5 截尾均值

对样本数据进行排序以后，去掉两端的部分极值，然后对剩下的数据求算术平均值，得到截尾均值。

用 trimmean 函数计算截尾均值。

● m = trimmean(X,percent) 剔除测量值中最大和最小 0.5%的数据以后，计算样本 ***X*** 的均值。截尾均值为样本位置参数的稳健性估计。若数据中有异常值，截尾均值为数据中心的一个代表性估计。若数据服从正态分布，则样本均值是比截尾均值更好的估计。

【例 3-5】 用蒙特卡罗法模拟正态数据的 10%截尾均值与样本均值之间的相关系数。

```
x = normrnd(0,1,100,100);
m = mean(x);
trim = trimmean(x,10);
sm = std(m);
strim = std(trim);
efficiency = (sm/strim).^2
efficiency =
   0.9702
```

3.3 描述离散趋势的统计量

描述离散趋势的统计量包括内四分极值、均值绝对差、极差、方差和标准差等。

3.3.1 内四分极值

内四分极值指的是样本数据的 75%与 25%位置处的值之差。

用 iqr 函数计算样本的内四分极值（IQR）。

● y = iqr(X) 计算 X 的内四分极值。IQR 是数据极差的稳健性估计。因为上下 25%的数据变化对其没有影响。

若数据中没有异常值，则 IQR 用于衡量数据的极差比标准差更具代表性。当数据源于正态分布时，标准差比 IQR 有效。常用 IQR*0.7413 来代替标准差。

【例 3-6】 用蒙特卡罗模拟来演示正态数据的 IQR 与样本标准差之间的相关系数。

```
x = normrnd(0,1,100,100);
s = std(x);
s_IQR = 0.7413 * iqr(x);
efficiency = (norm(s - 1)./norm(s_IQR - 1)).^2
efficiency =
   0.3297
```

3.3.2 均值绝对差

用 mad 函数计算数据样本的均值绝对差（MAD）。

● y = mad(X) 计算数据系列与取自该数据系列的样本均值之间绝对差的平均值。对于向量，mad(X)返回 X 的值的绝对平均值；对于矩阵，返回 X 的每一列的均值绝对差。

当数据取自正态分布时，均值绝对差用于数据极差估计的有效性比标准差要差一些。

可以用均值标准差乘以 1.3 来估计σ（正态分布的第 2 个参数）。

【例 3-7】 用蒙特卡罗模拟演示正态数据样本的均值绝对差与标准差之间的相关系数。

```
x = normrnd(0,1,100,100);
s = std(x);
s_MAD = 1.3 * mad(x);
efficiency = (norm(s - 1)./norm(s_MAD - 1)).^2
efficiency =
    0.5972
```

3.3.3 极差

极差指的是样本中最小值与最大值之间的差值。

用 range 函数计算样本的极差。

● y = range(X) 返回极差。对向量而言，range(X)为 X 中元素的极差。对矩阵而言，range(X)为包含每一列中元素极差的行向量。

用极差估计样本数据的范围具有计算简便的优点。缺点是异常值对它的影响较大，因此它是一个不可靠的估计值。

【例 3-8】 大样本标准正态分布随机数的极差近似为 6。下面首先生成 5 个包含 1000 个服从标准正态分布的随机数的样本，然后进行求极差的运算。

```
rv = normrnd(0,1,1000,5);
near6 = range(rv)
near6 =
    6.1451    6.4986    6.2909    5.8894    7.0002
```

3.3.4 方差

用 var 函数计算样本的方差。其调用格式和描述为：

● y = var(X) 计算 X 中数据的方差。对向量而言，var(X)为 X 中元素的方差。对于矩阵而言，var(X)是包含 X 中每一列元素方差的行向量,它通过除以 $n-1$ 来达到标称化，其中 n 为样本大小。对于正态分布数据，这使 var(x)成为σ^2 的最小方差无偏估计量。

● y = var(X, 1) 通过除以 n 来标称化并生成样本数据的二阶矩。

● y = var(X, w) 使用权重向量 w 计算方差。w 中元素的个数必须等于矩阵 X 的行数。对于向量 X，w 和 X 必须在长度上匹配。w 的每一个元素必须为正。

注意：令 SS 为 $\boldsymbol{X}$ 向量中元素与其均值之间的离差平方和，则 var(X) = $SS/(n-1)$为σ^2 的最小方差无偏估计量，var(X, 1) = SS/n 为σ^2 的最大似然估计量。

【例 3-9】

```
x = [-1 1];
w = [1 3];
v1 = var(x)
v1 =
     2
v2 = var(x,1)
v2 =
     1
v3 = var(x,w)
v3 =
    0.7500
```

3.3.5　标准差

样本数据 $x_1, x_2, ..., x_n$ 的标准差可以定义为

$$s = \left(\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 \right)^{\frac{1}{2}}$$

式中，样本均值为

$$\bar{x} = \frac{1}{n} \sum x_i$$

用 std 函数计算样本的标准差。

● y = std(X) 计算 X 中数据样本的标准差。对于向量 X，std(X) 为 X 中元素的标准差。对于矩阵，std(X)为包含 X 中每一列标准差的行向量。std(X)通过除以 n–1 来实现标称化，其中 n 为样本大小。对于正态分布数据，标准差的平方是σ^2的最小方差无偏估计量。

【例 3-10】 下面首先生成 6 列服从标准正态分布的随机数，每列有 100 个数。每一列中，标准差 y 的期望值均为 1。

```
x = normrnd(0,1,100,6);
y = std(x)
y =
   0.9536   1.0628   1.0860   0.9927   0.9605   1.0254
```

3.4　分组数据描述

利用 grpstats 函数计算分组综述统计量。

● means = grpstats(X, group) 根据 group 参数返回 X 的每一列的均值。X 为一观测量矩阵。group 为一正整数列，它指示 X 中每一行的分组关系。group 可以是向量、字符串数组或字符串的单元数组，也可以是包含一些分组变量（如{G1，G2，G3}）的单元数组。此时，如果个案观测值具有所有分组变量的公共值，则它们位于同一组中。

[means, sem, counts, name] = grpstats(x, group, alpha) 在 sem 中提供均值的标准误差。counts 为每组元素的个数，与其他输出具有相同的大小。每个组的名称包含在 name 中，当输入的分组值不是简单的分组个数时，它对于辨识和标注分组是很有用的。

grpstats(x, group, alpha) 绘制每一个均值的 100(1 - alpha)% 置信区间的图形。

【例 3-11】 下面有四组数据，每组数据有 100 个观测值。对于每组观测值，测量 5 个值，其真均值分别从 1 到 5。使用 grpstats 函数计算每个分组的均值。

```
group = unidrnd(4,100,1);
true_mean = 1:5;
true_mean = true_mean(ones(100,1),:);
x = normrnd(true_mean,1);
means = grpstats(x,group)
means =
   0.7947   2.0908   2.8969   3.6749   4.6555
   0.9377   1.7600   3.0285   3.9484   4.8169
   1.0549   2.0255   2.8793   4.0799   5.3740
   0.7107   1.9264   2.8232   3.8815   4.9689
```

3.5　包含缺失数据的样本描述

MATLAB 的统计工具箱中有一组名称以 nan 开头的函数，用于描述包含缺失数据的样本。表 3-1 示出了包含缺失数据的样本统计量的计算函数及其调用格式。

表 3-1　包含缺失数据的样本统计量

函 数 名 称	功　　能	调 用 格 式
nanmax	包含缺失数据，求样本数据的最大值	m=nanmax(X) [m,ndx]=nanmax(X) m=nanmax(a,b)
nanmin	包含缺失数据，求样本数据的最小值	m=nanmin(X) [m,ndx]=nanmin(X) m=nanmin(a,b)
nanmean	包含缺失数据，求样本数据的均值	y=nanmean(X)
nanmedian	包含缺失数据，求样本数据的中位数	y=nanmedian(X)
nanstd	包含缺失数据，求样本数据的标准差	y=nanstd(X)
nansum	包含缺失数据，求样本数据的和	y=nansum(X)

下面以 nanmax 函数为例介绍这一类函数的用法。

nanmax 函数计算包含缺失数据的样本数据的最大值。其调用格式和说明为：

- m = nanmax(a) 返回有效数据的最大值。NaN 表示缺失值。对于向量，nanmax(a)表示 a 的元素中最大的有效数据。对于矩阵，nanmax(a)表示包含每一列中有效数据的行向量。
- [m, ndx] = nanmax(a) 返回向量 ndx 中最大值的系数。
- m = nanmax(a, b) 返回 a 和 b 中的大者，a 和 b 必须具有相同的大小（指规模）。

注意：NaN 表示缺失值。缺失值与零不同，它表示对应的位置上没有观测值，不能简单地用 0 代替。

【例 3-12】

```
m = magic(3);
m([1 6 8]) = [NaN NaN NaN]
m =
   NaN     1     6
     3     5   NaN
     4   NaN     2
[nmax,maxidx] = nanmax(m)
nmax =
     4     5     6
maxidx =
     3     2     1
```

3.6　百分位数和图形描述

prctile 函数计算样本的百分位数。

● Y = prctile(X, p) 计算 X 中大于 p%的值，p 必须介于 0 和 100 之间。对于向量而言，prctile(X, p) 为 X 中元素的 p 百分位数。若 *p*=50，则 *Y* 为 ***X*** 的中值。对于矩阵 ***X*** 和标量 *p*，prctile(X, p)为包含每一列的 *p* 百分位数的行向量。若 ***p*** 为向量，则 ***Y*** 的第 *i* 行为 ***X*** 的 *p*(*i*)。

【例 3-13】

```
x = (1:5)'*(1:5)
x =
     1     2     3     4     5
     2     4     6     8    10
     3     6     9    12    15
     4     8    12    16    20
     5    10    15    20    25
y = prctile(x,[25 50 75])
y =
    1.7500    3.5000    5.2500    7.0000    8.7500
    3.0000    6.0000    9.0000   12.0000   15.0000
    4.2500    8.5000   12.7500   17.0000   21.2500
```

3.7 自助统计量

用 bootstrp 函数计算重复取样的自助统计量。其调用格式和描述为：

● bootstrp(nboot, 'bootfun', d1,...) 绘 nboot 个自助数据样本并用 bootfun 函数分析它们。nboot 必须为正整数。bootstrp 函数将数据 d1、d2 等传递给 bootfun 函数。

● [bootstat, bootsam] = bootstrap(...) 在 bootstat 参数中返回自助统计量。bootstat 的每一行包括对一个自助样本应用‘bootfun’得到的结果。若‘bootfun’返回一个矩阵，则该输出转换为一个长向量，以保存在 bootstat 中。bootsam 为一指数矩阵。

【例 3-14】 已知 15 个学生的 LSAT 分数和法学院 GPA。通过对这 15 个数据点进行重复采样，创建了 1000 个不同的数据集，计算每个数据集中两个变量之间的相关系数。

```
load lawdata
[bootstat,bootsam] = bootstrp(1000,'corrcoef',lsat,gpa);
bootstat(1:5,:)
ans =
    1.0000    0.3021    0.3021    1.0000
    1.0000    0.6869    0.6869    1.0000
    1.0000    0.8346    0.8346    1.0000
    1.0000    0.8711    0.8711    1.0000
    1.0000    0.8043    0.8043    1.0000
bootsam(:,1:5)
ans =
     4     7     5    12     8
     1    11    10     8     4
    11     9    12     4     2
    11    14    15     5    15
    15    13     6     6     2
     6     8     4     3     8
     8     2    15     8     6
    13    10    11    14     5
```

```
         1          7         12         14         14
         1         11         10          1          8
         8         14          2         14          7
        11         12         10          8         15
         1          4         14          8          1
         6          1          5          5         12
         2         12          7         15         12
    hist(bootstat(:,2))
```

生成直方图如图 3-1 所示。

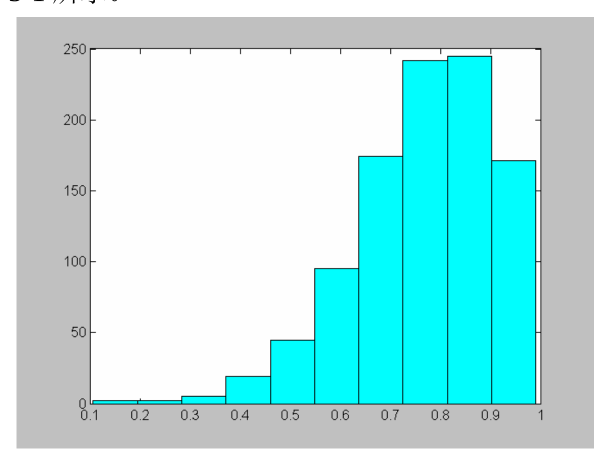


图 3-1　直方图

该直方图显示了覆盖整个自助样本的相关系数的变化。样本最小值为正表示 LSAT 和 GPA 之间的关系不是偶然的。

3.8　中心矩

k 阶中心矩可以用下式定义：

$$m_n = E(x - \mu)^k$$

式中，$E(x)$为 x 的期望。

moment 函数计算所有阶次的中心矩。

● m = moment(X, order) 返回由正整数 order 指定阶次的 X 的中心矩。对于向量，moment(X,order) 函数返回 X 的元素的指定阶次的中心矩。对于矩阵，moment(X, order)返回每一列的指定阶次的中心矩。

注意：一阶中心矩为 0，二阶中心矩为用除数 n（而非 n–1）得到的方差，其中 n 为向量 $\boldsymbol{X}$ 的长度或是矩阵 $\boldsymbol{X}$ 的行数。

【例 3-15】

```
    X = randn([6 5])
    X =
        1.1650    0.0591    1.2460   -1.2704   -0.0562
        0.6268    1.7971   -0.6390    0.9846    0.5135
        0.0751    0.2641    0.5774   -0.0449    0.3967
        0.3516    0.8717   -0.3600   -0.7989    0.7562
       -0.6965   -1.4462   -0.1356   -0.7652    0.4005
```

```
    1.6961   -0.7012   -1.3493    0.8617   -1.3414
m = moment(X,3)
m =
   -0.0282    0.0571    0.1253    0.1460   -0.4486
```

3.9 相关系数

用 corrcoef 函数计算样本数据的相关系数矩阵。

- R = corrcoef(X) 返回源于矩阵的相关系数矩阵，输入矩阵的行为观测量，列为变量。

相关系数矩阵 R 中的元素(i, j)与协方差矩阵 C（=cov(X)）的元素相对应，即有

$$R(i, j) = \frac{C(i, j)}{\sqrt{C(i,i)C(j, j)}}$$

3.10 协方差矩阵

用 cov 函数计算协方差矩阵。其语法格式为：

- C = cov(X)
- C = cov(X, Y)

cov 计算协方差矩阵。对于单一向量而言，cov(X)返回包含协方差的一个度量。对于行为观测量，列为变量的矩阵而言，cov(X)为协方差矩阵。

计算方差的函数 var(X)与 diag(cov(X))函数的作用相同。

计算标准差的函数 std(X)与 sqrt(diag(cov(X)))的作用相同。

cov(X, Y)的作用与 cov([X, Y])的相同，其中，X, Y 为长度相等的列向量。

cov 函数的算法为：

```
[n,p] = size(X);
X = X - ones(n,1) * mean(X);
Y = X'*X/(n-1);
```

3.11 峰度和偏度

3.11.1 峰度

样本的峰度由下式定义：

$$k = \frac{E(X-\mu)^4}{\sigma^4}$$

式中，$E(\boldsymbol{X})$为 $\boldsymbol{X}$ 的期望值。

峰度用于度量样本数据偏离某分布的情况，正态分布的峰度为 3。当样本数据的曲线峰值比正态分布的高时，峰度大于 3；反之，比正态分布的低时，峰度小于 3。

用 kurtosis 函数计算样本的峰度。

- k = kurtosis(X) 返回 X 的样本峰度。对于向量而言，kurtosis(X)函数为向量 X 中元素的峰度。对于矩阵而言，kurtosis(X)函数为 X 的每一列返回一个样本峰度。

注意：也有将峰度定义为计算值减 3 的，所以正态分布的峰度为 0。

【例 3-16】

```
X = randn([5 4])
X =
    1.1650    1.6961   -1.4462   -0.3600
    0.6268    0.0591   -0.7012   -0.1356
    0.0751    1.7971    1.2460   -1.3493
    0.3516    0.2641   -0.6390   -1.2704
   -0.6965    0.8717    0.5774    0.9846
k = kurtosis(X)
k =
   2.1658    1.2967    1.6378    1.9589
```

3.11.2 偏度

样本的偏度定义为

$$y = \frac{E(X-\mu)^3}{\sigma^3}$$

式中，$E(\boldsymbol{X})$为 $\boldsymbol{X}$ 的期望值。

偏度用于衡量样本均值的对称性，若偏度为负，则数据均值左侧的离散性比右侧的强；若偏度为正，则数据均值右侧的离散性比左侧的强。正态分布（或任何严格对称分布）的偏度为零。

用 skewness 函数计算样本偏度。

- y = skewness(X) 返回 X 的样本偏度。对于向量，skewness(x)为 X 的元素的偏度。对于矩阵，skewness(X)为包含每一列中样本偏度的行向量。

【例 3-17】

```
X = randn([5 4])
X =
    1.1650    1.6961   -1.4462   -0.3600
    0.6268    0.0591   -0.7012   -0.1356
    0.0751    1.7971    1.2460   -1.3493
    0.3516    0.2641   -0.6390   -1.2704
   -0.6965    0.8717    0.5774    0.9846
y = skewness(X)
y =
   -0.2933    0.0482    0.2735    0.4641
```

3.12 频数表

频数表中提供了各样本值出现的次数和百分比。

用 tabulate 函数绘制频数表。

- table = tabulate(X) 根据正整数向量 X 返回 table 矩阵。

table 的第 1 列包含 X 的值。第 2 列包含该值的实例个数。最后 1 列包含每个值的百分比。

不带输出参数的 tabulate 函数在命令窗口中显示一个格式化的表格。

【例 3-18】

```
tabulate([1 2 4 4 3 4])
  Value    Count   Percent
      1       1    16.67%
      2       1    16.67%
      3       1    16.67%
      4       3    50.00%
```

3.13 列联表

列联表常用于检验样本的独立性。用 crosstab 函数生成两个向量的列联表。

- table = crosstab(col1, col2) 使用两个正整数向量，返回一个矩阵 table。该矩阵的第 *ij* 个元素包含 col1 = *i* 且 col2 = *j* 时的所有实例的个数。
- [table, chi2, p] = crosstab(col1, col2) 还返回 χ^2 统计量 chi2，用于检验表中行和列的独立性。标量 p 为检验的显著性水平。当 p 值接近于 0 时，可以拒绝原假设，认为行和列之间是不独立的。

【例 3-19】 本例中首先生成两列包含 50 个服从离散均匀分布的随机数。第 1 列中的数为 1 到 3，第 2 列的为 1 和 2。试对这两列数据进行独立性检验。

```
r1 = unidrnd(3,50,1);r2 = unidrnd(2,50,1);
[table,chi2,p] = crosstab(r1,r2)
table =
    10    5
     8    8
     6   13
chi2 =
    4.1723
p =
    0.1242
```

结果 p=0.1242，说明上面的两列是相互独立的。也证明由该随机数生成器得到的随机数是随机的。

第 4 章　线 性 模 型

4.1　方差分析

事件的发生往往与多个因素有关，但各个因素对事件发生的影响可能是不一样的，而且同一因素的不同水平对事件发生的影响也是不同的。通过方差分析，便可以研究不同因素以及因素的不同水平对事件发生的影响程度。根据自变量个数的不同，方差分析可以分为单因子方差分析和多因子方差分析。

4.1.1　单因子方差分析

1．基本数学原理

一项试验有多个影响因素，如果只有一个在发生变化，则称为单因子分析。其基本原理为：假设某一试验有 s 个不同条件，则在每个条件（或称水平）下进行试验，可得到 s 个总体，分别记为 $\boldsymbol{X}_1$，$\boldsymbol{X}_2$，…，$\boldsymbol{X}_s$，各总体的平均数表示为 μ_1，μ_2，…，μ_s，各总体的方差表示为 σ_1^2，σ_2^2，…，σ_s^2。现在，在这 s 个总体服从正态分布且方差相等的情况下检验各总体的平均数是否相等，即检验假设 $H_0: \mu_1 = \mu_2 = \cdots = \mu_s$。当假设成立时，认为因素对试验结果之间没有显著影响。

观察值与总平均值之差的平方和称为离差平方和。进行单因子方差分析时，离差平方和被分解为组间平方和（也称条件误差，记为 $\mathrm{SS_A}$）和组内平方和（也称试验误差，记为 $\mathrm{SS_E}$）。对应地，总自由度 $\mathrm{d}f(=n-1)$ 被分解为组间自由度 $\mathrm{d}f_A(=s-1)$ 和组内自由度 $\mathrm{d}f_E(=n-s)$。

当零假设成立时，统计量

$$F = \frac{\mathrm{MS}_A}{\mathrm{MS}_E} = \frac{\mathrm{SS}_A / \mathrm{d}f_A}{\mathrm{SS}_E / \mathrm{d}f_E}$$

服从第 1 自由度为组间自由度，第 2 自由度为组内自由度的 F 分布。上式中 $\mathrm{MS_A}$ 称为组间均方，$\mathrm{MS_E}$ 称为组内均方。一般应有 $F \approx 1$。但如果得到的 F 值比 1 大得多，即条件误差比试验误差大得多，则条件（水平）不同起显著作用，因此，不能认为各总体的均值相同，故否定原假设。当 F 值小于 1 时，认为因素改变对实验结果引起的变动不显著，大部分试验误差由个体差异引起。

2．有关函数介绍

用 anova1 函数进行单因子方差分析。

● p = anova1(X) 进行平衡单因子方差分析，比较样本 $m \times n$ 的矩阵 X 中两列或多列数据的均值。其中，每一列包含一个具有 m 个相互独立观测值的样本。它返回 X 中所有样本取自同一群体（或取自均值相等的不同群体）的零假设成立的概率 p。若 p 值接近 0，则认为零假设可疑并认为列均值存在差异。

为了决定结果是否是“统计上显著”，需要确定 p 值。该值由自己确定。一般地，当 p 值

小于 0.05 或 0.01 时，认为结果是显著的。

anova1 函数还生成两个图形。第 1 个图为标准方差分析表，它将 ***X*** 中数据的误差分成两部分：

- 由于列均值的差异导致的误差（组间差）；
- 由于每一列数据与该列数据均值的差异导致的误差（组内差）。

方差分析表中有 6 列：

- 第 1 列显示误差的来源；
- 第 2 列显示每一个误差来源的平方和（SS）；
- 第 3 列显示与每一个误差来源相关的自由度(df)；
- 第 4 列显示均值平方和（MS），它是误差来源平方和与自由度的比值，即 SS/d*f*；
- 第 5 列显示 *F* 统计量，它是均值平方和的比值；
- 第 6 列显示 *p* 值，*p* 值是 *F* 的函数（fcdf）；当 *F* 增加时 *p* 值减小。

第 2 个图显示 X 的每一列的箱形图。箱形图中心线上较大的差异对应于较大的 *F* 值和较小的 *p* 值。

● anova1(X, group) 当 X 为矩阵时，利用 group 变量（字符数组或单元数组）作为 X 中样本的箱形图的标签。变量 group 中的每一行包含 X 中对应列中的数据的标签，所以 group 变量的长度必须等于 X 的列数。

当 ***X*** 为向量时，anova1 函数对 ***X*** 中的样本进行单因素方差分析，通过输入变量 group 进行标示。group 中的每个元素等价于 X 向量中的对应元素，所以，group 必须与 X 的长度相等。group 中包含的标签同样用于箱形图的标注。anova1 函数的向量输入形式不需要每个样本中的观测值个数相同，所以它适用于不平衡数据。

不必按顺序对样本进行标注。例如，若 X 包含 3 个不同温度（-27°,65°和 110°）上的观测值，可将这些数字作为 group 中的样本标签。若 group 中的一行包含一个空单元或空字符串，则该行和 X 中的对应观测值被忽略。每个输入中的空值（NaN）也同样被忽略。

● p = anova1(X, group, 'displayopt') 当'displayopt'参数设置为'on'(默认设置)时，激活 ANOVA 表和箱形图的显示；'displayopt'参数设置为'off'时，不予显示。

● [p, table] = anova1(...) 返回单元数组表中的 ANOVA 表（包含列标签和行标签）。（使用“Edit”菜单中的“Copy Text”选项可以将 ANOVA 表以文本形式复制到记事本中。）

● [p, table, stats] = anova1(...) 返回 stats 结构，用于进行多元比较检验。anova1 检验评价所有样本均值相等的零假设和均值不等的备择（候选）假设。有时进行检验，决定哪对均值差异显著，哪对差异不显著是很有效的。提供 stats 结构作为输入，使用 multcompare 函数可以进行此项检验。

注意：方差分析要求样本数据满足下面的假设条件：

① 所有样本数据满足正态分布条件；

② 所有样本数据具有相等的方差；

③ 所有观测值相互独立。

在基本满足前两个假设条件的情况下，一般认为 ANOVA 检验是稳健的。

3．应用实例

【例 4-1】 本例来自一项结构梁材料强度的研究。strength 向量衡量梁在 3 000 磅的压力作用下产生千分之几英寸的拉裂。强度更高的梁产生的拉裂较小。工程师们希望通过此项研

究确定该钢梁的强度是否与另外两种贵重得多的合金的强度相当。在向量中，钢用 1 代表，其他合金材料用 2 和 3 代表。

```
strength = [82 86 79 83 84 85 86 87 74 82 78 75 76 77 79 ...
            79 77 78 82 79];
alloy = {'st','st','st','st','st','st','st','st',...
         'al1','al1','al1','al1','al1','al1',...
         'al2','al2','al2','al2','al2','al2'};
```

%尽管在本例中对 alloy 向量进行了排序，但不必对分组变量进行排序。

```
p = anova1(strength,alloy)
p =
    1.5264e-04
```

p 值显示这 3 种金属具有显著差异。图 4-1 和图 4-2 分别为本问题的方差分析表和箱形图。箱形图用图形的形式进一步证明了钢梁比其他贵重合金的拉裂程度高一些。

Figure No. 1: One-way ANOVA

File Edit View Insert Tools Window Help

ANOVA Table

Source	SS	df	MS	F	Prob>F
Groups	184.8	2	92.4	15.4	0.0002
Error	102	17	6		
Total	286.8	19			

图 4-1　方差分析表

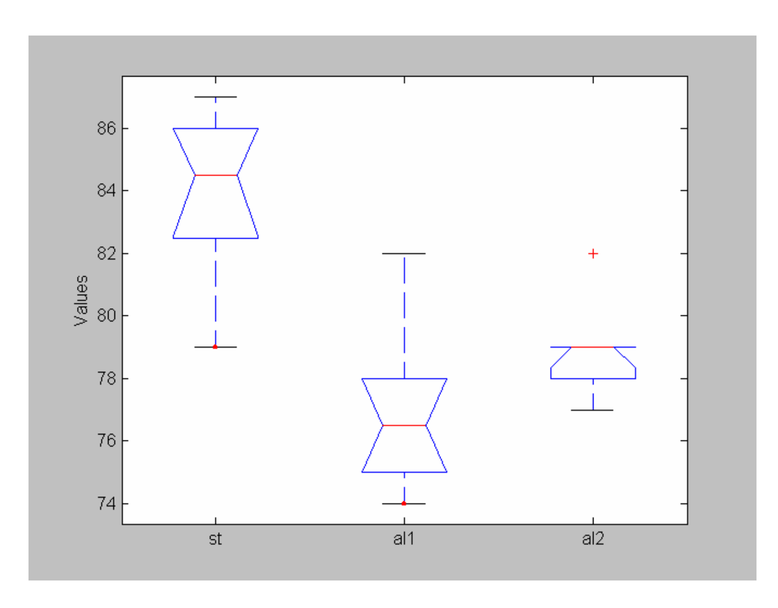


图 4-2　箱形图

【例 4-1】　一位教师想要检查 3 种不同的教学方法的效果，为此随机地选取了水平相当的 15 位学生。把他们分为 3 组，每组 5 人，每一组用一种方法教学，一段时间以后，这位教师给这 15 位学生进行统考，统考成绩见表 4-1。要求检验这 3 种教学方法的效果有没有显著差异（假设这 3 种教学方法的效果没有显著差异）。

表 4-1　学生统考成绩表

方　法	成　绩				
甲	75	62	71	58	73
乙	81	85	68	92	90
丙	73	79	60	75	81

```
score=[75 62 71 58 73;81 85 68 92 90;73 79 60 75 81]';
p=anova1(score)
p=
    0.0401
```

p 值小于 0.05，拒绝零假设，认为 3 种教学方法的效果存在显著差异。

图 4-3 中为本问题的方差分析表。从表中可以看出，$F_{0.05}(2, 12)$大于 F 为 4.26 的概率为 0.0401，小于 0.05，可以认为 $F_{0.05}(2, 12) < 4.26$。所以，在 0.05 的水平上可以认为 3 种教学方法的效果有显著差异。图 4-4 为本问题的箱形图，可见 3 种教学方法的效果还是存在显著差异。

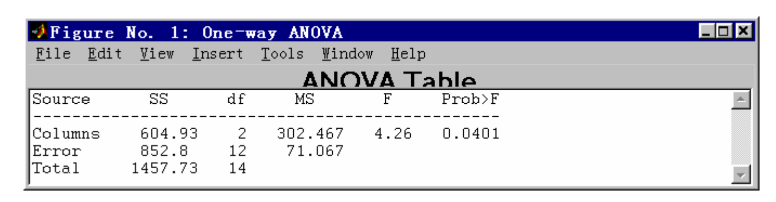


Figure No. 1: One-way ANOVA

File Edit View Insert Tools Window Help

ANOVA Table

Source	SS	df	MS	F	Prob>F
Columns	604.93	2	302.467	4.26	0.0401
Error	852.8	12	71.067		
Total	1457.73	14			

图 4-3　方差分析表

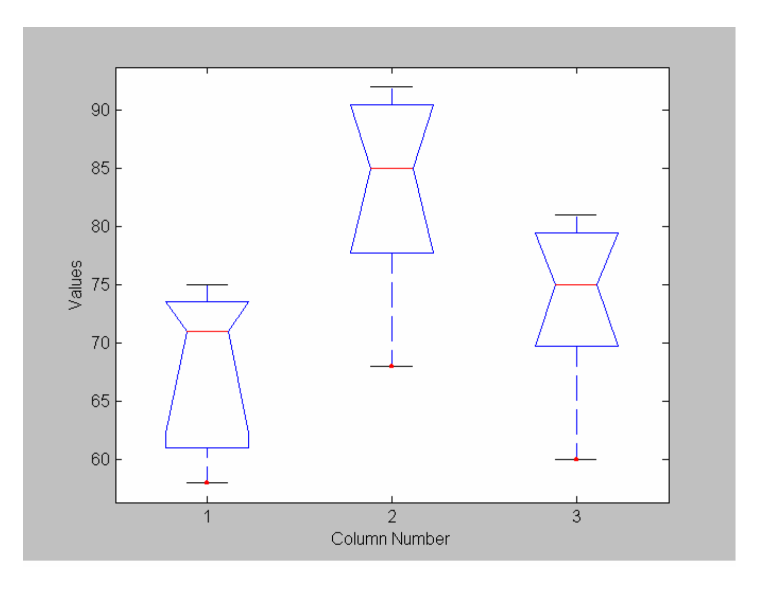


图 4-4　箱形图

4.1.2　双因子方差分析

1．基本数学原理

当有多个因素同时影响试验结果时，采用多因子方差分析。因素即变量，变量的不同取

值或者说因素的不同等级称为因素水平。

因素水平的改变所造成的试验结果的改变，称为主效应。当某一因素的效应随另一因素的水平不同而不同，则称这两个因素之间存在交互作用。由于交互作用引起的试验结果的改变称为交互效应。因素间是否存在交互效应，可以通过专门的数学方法进行检验。

进行多因子方差分析，还是需要对离差平方和进行分解。对于两个因素的情况，

（1）当没有交互作用时，离差平方和(SS)分解为

$$SS=SS_A+SS_B+SS_E$$

式中，SS_A 为 A 因素的离差平方和，SS_B 为 B 因素的离差平方和，SS_E 为误差平方和。

同样，自由度也要作相应的分解。A 因素对应的自由度为 $a-1$（a 为 A 因素的水平数），B 因素对应的自由度为 $b-1$（b 为 B 因素的水平数），误差项对应的自由度为 $n-a-b+1$（n 为试验次数）。

（2）当存在交互作用时，离差平方和分解为

$$SS=SS_A+SS_B+SS_{A\times B}+SS_E$$

式中，$SS_{A\times B}$为与因素A和因素B的交互效应对应的离差平方和。交互项的自由度为$(a-1)(b-1)$，误差项的自由度为 $n-ab$。

2．有关函数介绍

anova2 函数进行双因子方差分析。

● p = anova2(X, reps) 进行平衡双因子方差分析，以比较样本 X 中两列或两列以上和两行或两行以上数据的均值。不同列中的数据代表一个因子 A 的变化。不同行中的数据代表因子 B 的变化。

若在每一个行-列匹配点上有一个以上的观测值，则变量 reps 指示每一个单元中观测值的个数。下面的矩阵显示了一个列有两个水平，行有 3 个水平的构造格式，并且每个单元中有两个观测值（即 reps=2）。下标分别表示行、列和重复次数。

$$\begin{bmatrix} x_{111} & x_{111} \\ x_{112} & x_{121} \\ x_{211} & x_{122} \\ x_{212} & x_{221} \\ x_{311} & x_{222} \\ x_{312} & x_{322} \end{bmatrix}$$

当 reps=1（默认值）时，anova2 函数返回两个 p 值到 ***p*** 向量中：

- 零假设 H0A 的 p 值。零假设为源于因子 A 的所有样本（如 X 中的所有列样本）取自相同的总体。
- 零假设 H0B 的 p 值。零假设为源于因子 B 的所有样本（如 X 中的所有行样本）取自相同的总体。

当 reps>1 时，anova2 在 ***p*** 向量中返回第 3 个值：

- 零假设 H0AB 的 p 值。零假设为因子 A 和因子 B 之间没有交互效应。

如果任意一个 p 值接近于 0，则认为相关的零假设不成立。对于零假设 H0A，一个足够小的 p 值表示至少有一个列样本均值明显地不同于其他列样本均值，即因子 A 存在主效应。对于零假设 H0B，一个足够小的 p 值表示至少有一个行样本均值明显地不同于其他行样本均值，即因子 B 存在主效应。对于零假设 H0AB，一个足够小的 p 值表示因子 A 与因子 B 之间

存在交互效应。

为了决定结果是否是“统计上显著的”，需要确定 p 值。该值由自己确定。一般地，当 p 值小于 0.05 或 0.01 时，认为结果是显著的。

anova2 函数还显示一个含标准方差分析表的图形，它将 *X* 中数据的误差根据 reps 的值分为 3 部分或 4 部分：

- 由于列均值差异引起的误差；
- 由于行均值差异引起的误差；
- 由于行列交互作用引起的误差（如果 reps 大于它的默认值 1）；
- 剩下的误差为不能被任何系统因素解释的误差。

该方差分析表中包含 6 列：

- 第 1 列显示误差来源；
- 第 2 列显示源于每一个误差来源的平方和（SS）；
- 第 3 列为与每一个误差来源相关的自由度（d*f*）；
- 第 4 列为均值平方(MS)，它是误差平方和与自由度的比值，即 SS/d*f*；
- 第 5 列为 F 统计量，它是均值平方和的比值；
- 第 6 列为 p 值，它是 *F* 的函数(fcdf)；当 *F* 增加时 *p* 值减小。

● p = anova2(X, group, 'displayopt') 当'displayopt'参数设置为'on'(默认设置)时，激活 ANOVA 表和箱形图的显示；'displayopt'参数设置为'off'时，不予显示。

● [p, table] = anova2(...) 返回单元数组表中的 ANOVA 表（包含列标签和行标签）。（使用 Edit 菜单中的 Copy Text 选项可以将 ANOVA 表以文本形式复制到记事本中。）

● [p, table, stats] = anova2(...) 返回 stats 结构，用于进行多元比较检验。anova2 检验评价所有行、列和交互效应相等的零假设和它们不等的备择假设。有时进行检验，决定哪对均值显著差异，哪对差异不显著是很有效的。将 stats 结构作为输入，使用 multcompare 函数可以进行此项检验。

3．应用实例

【例 4-3】 下面的数据来源于一项爆玉米花品牌与爆玉米花方法类型的研究。popcorn 的列对应品牌 (包括 Gourmet, National 和 Generic)，行对应方法类型(Oil 和 Air)。该研究中对每一个品牌的爆玉米花用每一种方法爆了 3 次。生成的值显示在装爆玉米花的杯上。

```
load popcorn
popcorn
popcorn =
    5.5000    4.5000    3.5000
    5.5000    4.5000    4.0000
    6.0000    4.0000    3.0000
    6.5000    5.0000    4.0000
    7.0000    5.5000    5.0000
    7.0000    5.0000    4.5000
p = anova2(popcorn,3)
p =
    0.0000    0.0001    0.7462
```

向量 ***p*** 显示 3 个品牌爆玉米花的 p 值为 0.0000，两个爆玉米花方法类型的 p 值为 0.0001，品牌与方法之间的交互作用的 p 值为 0.7462。这些值说明品牌和方法对产出都有影响，但二

者之间没有交互作用。图 4-5 所示的方差分析表也显示了该结论。

Figure No. 1: Two-way ANOVA

File Edit View Insert Tools Window Help

ANOVA Table

Source	SS	df	MS	F	Prob>F
Columns	15.75	2	7.875	56.7	0
Rows	4.5	1	4.5	32.4	0.0001
Interaction	0.0833	2	0.04167	0.3	0.7462
Error	1.6667	12	0.13889		
Total	22	17			

图 4-5　双因素方差分析表

【例 4-4】　为了考察 4 种不同燃料与 3 种不同型号的推进器对火箭射程（单位：海里）的影响，做了 12 次试验，得数据如表 4-2 所示。

表 4-2　燃料-推进器-射程数据表

	推进器 1	推进器 2	推进器 3
燃料 1	58.2	56.2	65.3
燃料 2	49.1	54.1	51.6
燃料 3	60.1	70.9	39.2
燃料 4	75.8	58.2	48.7

要求分析燃料和推进器的不同是否对火箭的射程有显著影响。零假设为没有影响。

```
disp=[58.2 56.2 65.3;49.1 54.1 51.6;60.1 70.9 39.2;75.8 58.2 48.7]';
p=anova2(disp,1)
   p =
      0.7387    0.4491
```

由于燃料和推进器对应的 p 值均大于 0.05，所以可以接受零假设 H_{01} 和 H_{02}，认为燃料和推进器对火箭的射程没有显著影响。图 4-6 为本问题的方差分析表。

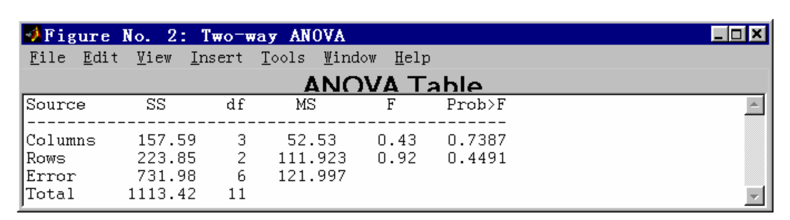


Figure No. 2: Two-way ANOVA

File Edit View Insert Tools Window Help

ANOVA Table

Source	SS	df	MS	F	Prob>F
Columns	157.59	3	52.53	0.43	0.7387
Rows	223.85	2	111.923	0.92	0.4491
Error	731.98	6	121.997		
Total	1113.42	11			

图 4-6　双因素方差分析表

【例 4-5】　设火箭的射程在其他条件基本相同时与燃料种类及推进器型号有关。现在考虑 4 种不同的燃料及 3 种不同型号的推进器，对于每种搭配各发射了火箭两次，得数据如表 4-3 所示。

表 4-3　燃料-推进器-射程数据表

	推进器 1	推进器 2	推进器 3
燃料 1	58.2	56.2	65.3
	52.6	41.2	60.8
燃料 2	49.1	54.1	51.6
	42.8	50.5	48.4

续表

	推进器 1	推进器 2	推进器 3
燃料 3	60.1 58.3	70.9 73.2	39.2 40.7
燃料 4	75.8 71.5	58.2 51.0	48.7 41.4

要求检验各自变量和自变量的交互效应是否对火箭的射程有显著影响。零假设为没有影响。

```
disp2=[58.2 52.6 49.1 42.8 60.1 58.3 75.8 71.5
        56.2 41.2 54.1 50.5 70.9 73.2 58.2 51.0
       65.3 60.8 51.6 48.4 39.2 40.7 48.7 41.4]';
anova2(disp2,2)
    ans =
         0.0035    0.0260    0.0001
```

由 ans 向量可知，燃料、推进器和二者交互效应对应的 p 值分别为 0.0035、0.0260 和 0.0001。三者均小于0.05，所以拒绝 3 个零假设，认为燃料、推进器和二者的交互效应对于火箭的射程都是有显著影响的。图 4-7 为本问题的方差分析表。

```
Figure No. 1: Two-way ANOVA
File  Edit  View  Insert  Tools  Window  Help
                         ANOVA Table
Source             SS      df     MS         F      Prob>F
-----------------------------------------------------------
Columns          370.98     2   185.49      9.39   0.0035
Rows             261.68     3    87.225     4.42   0.026
Interaction     1768.69     6   294.782    14.93   0.0001
Error            236.95    12    19.746
Total           2638.3     23
```

图 4-7　方差分析表

4.1.3　多因素方差分析

基本原理可以参见前面双因子方差分析的内容。

用 anovan 函数进行 N 因素方差分析。

- p = anovan(X, group) 进行平衡或不平衡数据的多因素方差分析，比较向量 X 中相对于 *N* 个不同因子的观测值的均值。X 中观测值的因子和因子水平由单元数组 group 指定。group 中 *N* 个单元中的每一个包含一系列因子水平，确定相对于 *N* 个因子中某一个的 X 的观测值。每个单元中的列表可以是向量、字符数组或字符串单元数组，并且必须与 X 具有相同的元素个数。

作为一个实例，考虑下面的 X 和 group 输入：

```
X = [x1 x2 x3 x4 x5 x6 x7 x8];
group = {[1 2 1 2 1 2 1 2];...
        ['hi';'hi';'lo';'lo';'hi';'hi';'lo';'lo'];...
        {'may' 'may' 'may' 'may' 'june' 'june' 'june' 'june'}};
```

本例中，anovan(X,group)为三因素方差分析，每个因子具有两个水平。X 中的每个观测值由 group 中的因子水平的组合确定。若因子为 A，B 和 C，则观测值 x1 与下面的因素有关：

- A 因子的水平 1；
- 因子 B 的水平'hi'；
- 因子 C 的水平'may'。

近似地，观测值 x6 与下面因素有关：

- 因子 A 的水平 2；
- 因子 B 的水平'hi'；
- 因子 C 的水平'june'。

输出向量 p 包含 *N* 个主效应零假设的 p 值。元素 p(1)包含零假设 H0A 的 p 值，H0A 假设因子 A 所有水平上的样本都取自相同总体；元素 p(2)包含零假设 H0B 的 p 值，H0B 假设因子 B 所有水平上的样本都取自相同总体；照此类推。

若任何一个 p 值接近于 0，则可以怀疑相关零假设是否成立。例如，对于 H0A 假设，一个足够小的 *p* 值表示至少有一个 A 样本均值与其他 A 样本均值有差异，即，有源于因子 A 主效应。为了决定结果是否是“统计上显著的”，需要确定 p 值。该值由用户朋友自己确定。一般地，当 p 值小于 0.05 或 0.01 时，认为结果是显著的。

anovan 函数还生成一个图形，显示标准方差分析表。默认时，将 X 中数据的变异性分为：

- 模型中由每个因子水平之间的差异导致的变异（可解释）；
- 不能被任何系统因素解释的变异。

方差分析表共有 6 列，具体内容与因子方差分析表相同。

● p = anovan(X, group, 'model') 用'model'指定的模型进行方差分析，其中，'model'可以是'linear', 'interaction', 'full'，一个整数或向量。

- 'linear'模型为默认选项。只计算 *N* 个主效应零假设的 p 值。
- 'interaction'模型计算 *N* 个主效应和二因子交互效应零假设的 p 值。
- 'full'模型计算 *N* 个主效应和所有水平交互效应零假设的 p 值。
- 整数值 *k* （*k*≤*N*)：anovan 函数计算所有第 *k* 水平交互效应零假设的 p 值。当 *k*=1 时等价于’linear’模型；当 *k*=2 时，等价于'interaction'模型；当 *k*=*N* 时，等价于'full'模型。
- 向量：控制主效应和交互项更简洁的方式，’model’可以指定一个在方差分析模型中为每一个主效应或交互项包含一个元素的向量。

表 4-4 示出三因子方差分析的编码。

例如，如果'model'为向量[2 4 6]，则输出向量 ***p*** 包含基于主效应 B，C 和交互效应 BC 零假设的 p 值。

● p = anovan(X, group, 'model', sstype) 用根据 sstype 指定的平方和类型进行方差分析，sstype 可以取 1，2 或 3，分别指示类型 1，类型 2 或类型 3。默认时为类型 3。sstype 的取值只影响非平衡数据的计算。

任意项的平方和通过比较两个模型来确定。项目的第 1 类平方和是通过在已包含前面列出项目的拟合模型中添加项目得到的残差平方和的减小量。第 2 类平方和是通过在不包含前面列出项目的拟合模型中添加项目得到的残差平方和的减小量。第 3 类平方和是通过在包含所有其他项目（但它们的效应必须遵守“σ 约束”以使模型可以估计）的拟合模型中添加项目得到的残差平方和的减小量。假设现在用两个因子和它们

表 4-4　三因子方差分析编码表

3 位编码	十进制数值	对应的方差分析项
[0 0 1]	1	主项 A
[0 1 0]	2	主项 B
[1 0 0]	4	主项 C
[0 1 1]	3	交互项 AB
[1 1 0]	6	交互项 BC
[1 0 1]	5	交互项 AC
[1 1 1]	7	交互项 ABC

的交互项拟合一个模型，并且项目按 A，B 和 AB 的顺序出现。令 R(•)代表模型的残差平方和，如 R(A, B, AB)是拟合整个模型的残差平方和，R(A)是拟合 A 的主效应的残差平方和，R(1)为拟合均值的残差平方和。3 种类型的平方和如表 4-5 所示。

表 4-5　3 种类型的平方和

项　目	第 1 类平方和	第 2 类平方和	第 3 类平方和
A	R(1)-R(A)	R(B)-R(A, B)	R(B,AB)-R(A, B, AB)
B	R(A)-R(A, B)	R(A)-R(A, B)	R(A, AB)-R(A, B, AB)
AB	R(A, B)-R(A, B, AB)	R(A, B)-R(A, B, AB)	R(A, B)-R(A, B, AB)

第 3 类平方和的模型添加了σ 约束。这表示，例如，在拟合 R(B, AB)时，对于 B 的每个元素，在 A 上（或对于 A 的每个值，在 B 上），AB 效应数组的和为 0。

● p=anovan(X, group, 'model', sstype, gnames) 用字符串数组 gnames 中的字符串值标注方差分析表中 N 个试验因子。数组可以是一个字符串矩阵，每个观测值对应一行；或字符串单元数组，每个观测值对应一个元素。

当没有指定 gnames 时，使用默认标签'X1'，'X2'，'X3', ..., 'XN'。

● p=anovan(X, group, 'model', sstype, gnames, 'displayopt') 当'displayopt'参数设置为'on'(默认设置)时，激活 ANOVA 表和箱形图的显示；'displayopt'参数设置为'off'时，不予显示。

● [p,table] = anovan(...) 返回单元数组表中的 ANOVA 表（包含列标签和行标签）。（使用 Edit 菜单中的 Copy Text 选项可以将 ANOVA 表以文本形式复制到记事本中。）

● [p, table, stats] = anovan(...) 返回 stats 结构，用于进行多元比较检验。anovan 检验评价某因子（或某项）不同水平的效应相同的零假设和它们不相同的备择假设。有时进行检验，决定哪对均值显著差异，哪对差异不显著是很有效的。提供 stats 结构作为输入，使用 multcompare 函数可以进行此项检验。

● [p, table, stats, terms] = anovan(...) 返回方差分析计算中用到的主项和交互项。项目用与上面输入’model’相同的格式在输出向量项中进行编码。当'model'本身用该向量格式指定以后，项目中返回的向量是等价的。

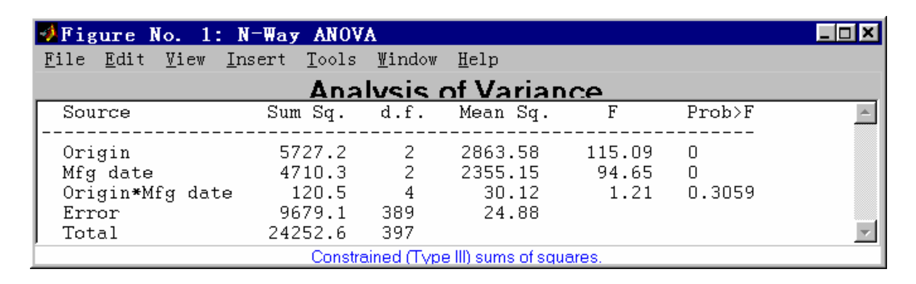
【例 4-6】 前面使用 anova2 函数分析了平衡设计中某响应上二因子的效应。对于不平衡数据，我们用 anovan 函数来分析。

数据集 carbig 包含 406 辆小汽车的测量数据。现在研究小汽车的运行指标 mileage（哩数）与汽车产地和出产时间的关系。

```
load carbig
anovan(MPG,{org when},2,3,{'Origin';'Mfg date'})
ans =
          0
          0
     0.3059
```

交互项的 p 值大于 0.05，表示出产时间与产地之间基本上没有交互效应。而汽车产地和出产时间对应的 p 值均为 0，说明二者对于运行指标 mileage 具有显著的影响。

图 4-8 为方差分析表。可参照描述部分进行阅读，不再重复。


Figure No. 1: N-Way ANOVA

File Edit View Insert Tools Window Help

Analysis of Variance

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
Origin	5727.2	2	2863.58	115.09	0
Mfg date	4710.3	2	2355.15	94.65	0
Origin*Mfg date	120.5	4	30.12	1.21	0.3059
Error	9679.1	389	24.88		
Total	24252.6	397			

Constrained (Type III) sums of squares.

图 4-8　多因素方差分析表

4.1.4　方差分析工具

用 aoctool 函数生成进行方差分析模型拟合和预测的交互图。

aoctool(x,y,g) 对于 g 数组中定义的列向量 x 和 y，分别用直线进行拟合。这些类型的模型即后面要讲到的单因子多元方差分析模型（ANOCOVA）。输出包括数据和预测曲线的交互图、方差分析表和参数估计表。

（1）用图形改变模型并检验模型的不同部分。

● aoctool(x, y, g, alpha) 确定预测区间的置信水平。置信水平为 100*(1-alpha)%。alpha 的默认值为 0.05。

● aoctool(x, y, g, alpha, xname, yname, gname) 指定在图中和表中使用 x，y 和 g 变量的名称。若输入 x，y 和 g 变量的名称，则 aoctool 函数直接使用这些名称。若输入这些变量中某一个的表达式，则可以通过提供这些变量在表达式处指定使用名称。例如，若输入 m(:, 2)作为 x 变量，则可以选择输入'Col 2'作为 xname 变量。

● aoctool(x, y, g, alpha, xname, yname, gname, 'displayopt') 通过将'displayopt'参数设置为'on'来激活图形和表格的显示。将该参数设置为’off’，则取消显示。

● aoctool(x, y, g, alpha, xname, yname, gname, 'displayopt', 'model') 指定进行拟合的初始模型。’model’的值可以是下面字符串中的一个：

- 'same mean'——忽略分组，拟合单个均值；
- 'separate means'——对每个组拟合单均值；
- 'same line'——忽略分组，拟合单条直线；
- 'parallel lines'——为每个组拟合单条直线，但要求直线必须平行；
- 'separate lines'——为每个组拟合单条直线，没有约束；
- h = aoctool(...)——返回图中直线对象的句柄向量。

● [h, atab, ctab] = aoctool(...) 返回包含方差分析表(atab)和系数估计表(ctab)中入口的单元数组。（可以使用 Edit 菜单中的 Copy Text 选项将表格以文本方式拷贝到记事本中。）

● [h, atab, ctab, stats] = aoctool(...) 返回一个 stats 结构，以进行多元比较检验。方差分析表输出包括斜率或截距相等的假设检验。有时，进行检验以决定哪些数据对有显著差异，哪些数据对没有显著差异是很有效的。可以通过提供 stats 结构作为输入，使用 multcompare 函数进行检验。可以检验斜率、截距或总体边缘均值。

【例 4-7】 本例演示怎样无交互地拟合不同模型。首先，装载更小的汽车数据集，并拟合一个斜率模型，然后进行系数估计。

```
[h,a,c,s] = aoctool(Weight,MPG,Model_Year,0.05,...
```

```
                        '','','','off','separate lines');
c(:,1:2)
ans =
    'Term'            'Estimate'
    'Intercept'       [45.97983716833132]
    '  70'            [-8.58050531454973]
    '  76'            [-3.89017396094922]
    '  82'            [12.47067927549897]
    'Slope'           [-0.00780212907455]
    '  70'            [ 0.00195840368824]
    '  76'            [ 0.00113831038418]
    '  82'            [-0.00309671407243]
```

概略地讲，MPG 与 Weight 之间相关曲线的截距接近于 45.98，斜率接近-0.0078。每个组的系数都有一定程度的偏离，如 1970 年生产的汽车数据的截距为 45.98-8.58 = 37.40。

然后，我们用平行线进行拟合。

```
[h,a,c,s] = aoctool(Weight,MPG,Model_Year,0.05,...
                        '','','','off','parallel lines');
c(:,1:2)
ans =
    'Term'            'Estimate'
    'Intercept'       [43.38984085130596]
    '  70'            [-3.27948192983761]
    '  76'            [-1.35036234809006]
    '  82'            [ 4.62984427792768]
    'Slope'           [-0.00664751826198]
```

这里，对于每个组都赋予了单独的交互项，但斜率必须是相同的。

（2）用 Dummyvar 函数生成二项变量（0-1）矩阵。该函数的调用格式为：

- D = dummyvar(group) 生成 0-1 列矩阵 D。group 的每一列包含一些正整数，指示单行的分组关系。

【例 4-8】 假设我们正在研究某个生产过程中两台机器与 3 个操作工人的生产效果。group 的第 1 列将根据使用的机器来决定赋值 1 或 2。group 的第 2 列根据操作人员的不同赋值 1，2 或 3。

```
group = [1 1;1 2;1 3;2 1;2 2;2 3];
D = dummyvar(group)
D =
     1     0     1     0     0
     1     0     0     1     0
     1     0     0     0     1
     0     1     1     0     0
     0     1     0     1     0
     0     1     0     0     1
```

4.2 线性回归

在实际生活中，某个现象的发生或某种结果的得出往往与其他某个或某些因素有关，但这种关系又不是确定的，只是从数据上可以看出有“有关”的趋势。回归分析就是用来研究

具有这种特征的变量之间的相关关系的。

线性回归假设因变量与自变量之间为线性关系，用一定的线性回归模型来拟合因变量和自变量的数据，并通过确定模型参数来得到回归方程。根据自变量的多少，线性回归可有不同的划分。当自变量只有一个时，称为一元线性回归；当自变量有多个时，称为多元线性回归。另外，可以转化为一元线性回归的非线性回归问题也在本节中一并介绍。

4.2.1　基本数学原理

1．一元线性回归

（1）回归模型与参数的确定

一元线性回归研究因变量与一个自变量之间的线性关系。其回归模型为

$$y=b_0+b_1x$$

式中，y 为因变量，x 为自变量，b_0，b_1 为待定参数（b_0 称为常数项，b_1 称为回归系数）。

通常采用最小二乘法来确定上面两个待定参数，即要求观测值与利用上面回归模型得到的拟合值之间差值的平方和最小。差值平方和达到最小时的模型参数便作为待定参数的最终取值，代入模型，便可以确定回归方程。

（2）回归系数的显著性检验

给定以上模型和实测数据以后，总可以得到待定参数的拟合值，但由此确定的回归方程式不一定有意义。因此，需要对得到的回归系数做显著性检验，即检验回归系数是否为 0，如果为 0。则说明因变量与自变量无关，回归方程无意义。回归系数的显著性检验有多种方法，下面介绍 F 检验法、t 检验法和相关系数检验法。

① F 检验法：为了对回归方程做显著性检验，首先将观测值和拟合值差值的平方和(SS)分解为回归平方和(SS_R)和残差平方和(SS_E)，用以下统计量进行检验：

$$F=\frac{\mathrm{SS}_R}{\mathrm{SS}_E/(n-2)}$$

式中，n 为数据组数。当 F 值大于一定的临界值时，拒绝原假设，认为因变量与自变量之间是相关的。

② t 检验法：做 t 检验时取下面的统计量

$$T=\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\,\frac{b_1}{\sigma}$$

当该统计量大于一定的临界值时，拒绝原假设，认为因变量与自变量之间是相关的。

③ 相关系数检验法：做相关系数检验时，取下面的统计量

$$R=\frac{\sum_{i=1}^{n}(x_i-\bar{x})(Y_I-\bar{Y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2\sum_{I=1}^{N}(Y_i-\bar{Y})^2}}$$

式中，R 称为相关系数。当相关系数的绝对值大于一定的临界值时，拒绝原假设。

（3）回归系数的区间估计

由最小二乘法得到的是回归系数的点估计（称为最小二乘估计），实际问题中常要求给出回归系数的置信区间。常数项和回归系数的置信水平为 $1-\alpha$ 的置信区间可由下面两式给出

$$(b_0 - t_{1-\frac{\alpha}{2}}(n-2) \cdot \sigma_{b_0}, b_0 + t_{1-\frac{\alpha}{2}}(n-2) \cdot \sigma_{b_0})$$

$$(b_1 - t_{1-\frac{\alpha}{2}}(n-2) \cdot \sigma_{b_1}, b_1 + t_{1-\frac{\alpha}{2}}(n-2) \cdot \sigma_{b_1})$$

（4）预测

经检验回归系数为显著的以后，便可利用回归方程式做预测。只要输入自变量的一个取值，便可获得一个因变量的估计值。当给定预测精度时，就可获得回归系数的预测区间。

（5）假设的检验

进行线性回归时，有 4 个基本假定，即因变量与自变量之间线性关系的假定、残差的独立性假定、残差的方差齐性假定和残差正态分布的假定。在实际工作中应对这些假定逐一进行检验，对于不符合假定的，应采取相应的措施进行处理。

① 线性诊断：对于一元线性回归问题，直接做自变量与因变量的散点图便可大致地看出它们之间是否具有线性关系。另外，利用残差图也可以进行判断。在标准残差-标准预测值散点图中的各点应在纵坐标零点对应的直线上下比较均匀地分布，而不呈现一定的规律。

② 残差的独立性诊断：可以在运行过程时保存残差，然后对保存的残差变量用 Durbin-Watson 检验法进行诊断。该检验法采用的统计量为

$$\mathrm{DW} = \frac{\sum_{i=2}^{n}(e_i - e_{i-1})^2}{\sum_{i=1}^{n} e_i^2}$$

式中，e_i 为当前点的残差，e_{i-1} 为前一点的残差，n 为数据组数。

当$|\mathrm{DW}-2|$过大时拒绝原假设，认为相邻两点的残差之间是相关的。当 DW<2 时，认为相邻两点的残差为正相关，当 DW>2 时，认为相邻两点的残差为负相关，只有当 DW≈2 时，认为相邻两点的残差之间是相互独立的。

③ 残差的方差齐性诊断：这可以通过生成和分析标准化预测值-学生化残差散点图来实现。当图中各点分布没有明显的规律性，即残差的分布不随预测值的变化而增大或减小时，认为残差是方差齐性的。当然，也可以通过保存残差对保存变量做方差齐性诊断。

④ 残差的正态性诊断：这可以通过直方图和 P-P 正态概率图来实现。

2．多元线性回归

多元线性回归的回归模型为

$$y=b_0+b_1x_{i1}+b_2x_{i2}+\cdots b_nx_{in},\quad i=1, 2, \cdots, n$$

模型中各系数与常数项通常还是利用最小二乘法来求得。与一元线性回归一样，进行多元线性回归还是需要进行回归系数的检验，估计回归系数的置信区间，进行预测与假设检验等方面的讨论。研究的方法和思路与前面大同小异，可以参见一元回归的内容。根据多元回归时自变量选择的不同，多元回归可以有多种不同的计算方法，下面分别予以叙述。

（1）全回归法

进行全回归时，所有的自变量进入回归方程。使用这种方法，一般具有较高的回归系数，但一些对因变量没有显著影响的自变量也可能进入回归方程。

（2）向前法

该方法比较所有自变量与因变量的偏相关系数，然后选择最大的一个做回归系数显著性检验，决定其是否进入回归方程。这种方法的缺点是某自变量选入方程以后，就一直留在方程中，不再剔除（因此该法又称为“只进不出法”）。然而在较早阶段进入回归方程的，当时认为最好的变量在较晚阶段可能因为它与方程中其他变量之间的相互关系而显得不再重要，因而有剔除的必要。

（3）向后法

与向前法相反，向后法又称为“只出不进法”。该法首先计算包含所有变量的回归方程，然后用偏 F 检验逐个剔除对因变量无显著影响的自变量，直到每一个自变量在偏 F 检验下都有显著性结果为止。该法得到的结果比全回归法的简洁，但有一个缺点，即变量在向后消元过程中如果被剔除，它将永远不会在方程中重新出现。然而一个被剔除的变量能可能在其他变量被剔除以后又对因变量有显著影响。

（4）逐步回归法

逐步回归法是对向前法的改进。它首先对偏相关系数最大的变量做回归系数显著性检验，以决定该变量是否进入回归方程。然后对方程中的每个变量作为最后选入方程的变量求出偏 F 值，对偏 F 值最小的那个变量做偏 F 检验，决定它是否留在回归方程中。重复此过程，直至没有变量被引入，也没有变量可剔除时为止。这样，应用逐步回归法时，既有引入变量也有剔除变量，原来被剔除的变量在后面又可能被引入到回归方程中来。它是目前应用较为广泛的一种多元回归方法。

4.2.2　有关函数介绍

1．rcoplot 函数

用 rcoplot 函数绘残差个案排序图。

● rcoplot(r,rint) 在回归分析得到的残差处显示一个置信区间的误差条图。图中的残差按案例号排序。r 和 rint 是 regress 函数的输出参数。

【例 4-9】生成图 4-9，图中用误差条图显示残差的 95%置信区间。所有的误差条通过零线时，表示数据中没有异常值。所以以下数据中没有异常值。

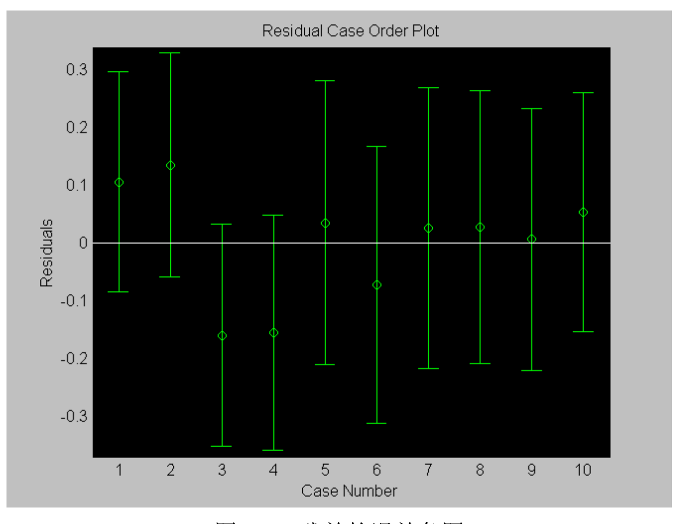


图 4-9　残差的误差条图

```
X = [ones(10,1) (1:10)'];
y = X * [10;1] + normrnd(0,0.1,10,1);
[b,bint,r,rint] = regress(y,X,0.05);
rcoplot(r,rint);
```

2. regress 函数

用 regress 函数进行多元线性回归。

- b = regress(y, X) 返回 X 处 y 的最小二乘拟合值。该函数求解线性模型：

$$y = X\beta + \varepsilon$$

$$\varepsilon \sim N(0, \sigma^2 I)$$

式中：

$\boldsymbol{y}$ 为一 $n \times 1$ 的向量，$\boldsymbol{X}$ 为一 $n \times p$ 的矩阵，β 为一 $p \times 1$ 的参数向量，ε 为一 $n \times 1$ 的随机分布向量。

- [b, bint, r, rint, stats] = regress(y, X) 在 b 中返回 β 的估计，在 $p \times 2$ 的向量 b 中返回 β 的 95%置信区间。r 中为残差，在 $n \times 2$ 的向量 rint 中返回每一个残差的 95%置信区间。stats 向量包含 R^2 统计量、回归的 F 值和 p 值。

- [b, bint, r, rint, stats] = regress(y, X, alpha) 给出 bint 和 rint 的 100(1-alpha)% 置信区间。例如，当 alpha = 0.2 时，给出 80%置信区间。

【例 4-10】 假设真实模型为

$$y = 10 + x + \varepsilon$$

$$\varepsilon \sim N(0, 0.01I)$$

式中 $\boldsymbol{I}$ 为等价矩阵。

```
X = [ones(10,1) (1:10)']
X =
    1    1
    1    2
    1    3
    1    4
    1    5
    1    6
    1    7
    1    8
    1    9
    1   10
y = X * [10;1] + normrnd(0,0.1,10,1)
y =
  11.1165
  12.0627
  13.0075
  14.0352
  14.9303
  16.1696
  17.0059
  18.1797
  19.0264
  20.0872
```

```
[b,bint] = regress(y,X,0.05)
b =
  10.0456
   1.0030
bint =
   9.9165  10.1747
   0.9822   1.0238
```

3．leverage 函数

该函数生成回归的中心化杠杆值。

● h = leverage(DATA) 找到 DATA 矩阵中每一行的中心化杠杆值，DATA 为线性回归模型。

● h = leverage(DATA, 'model') 找到回归的中心化杠杆值，并使用指定的模型类型。'model' 可以是下面字符串中的一个：

'interaction'——包含常数项、线性项和交互项。

'quadratic'——交互项加上平方项。

'purequadratic'——包含常数项、线性项和平方项。

中心化杠杆值用于衡量由于给定观测值在输入空间中的位置而引起的对回归的影响。

【例 4-11】 一条较好的规则是将中心化杠杆值与 $2p/n$ 进行比较，其中，n 为观测量的个数，p 为模型中的参数个数。对于 Hald 数据集，该值为 0.7692。

```
load hald
h = max(leverage(ingredients,'linear'))
h =
   0.7004
```

因为 0.7004 < 0.7692，所以根据本规则，没有高的中心化杠杆值点（即强影响点）。

4．regstats 函数

用 regstats 函数回归诊断图形用户界面。

● regstats(responses, DATA) 为带常数项的线性模型进行回归诊断。因变量为 responses 向量。自变量的值在 DATA 矩阵中。该函数创建一个图形窗口，其中提供了一组用指定变量名将诊断统计量保存到基本工作空间的核选框。

● regstats(responses, data, 'model') 控制回归模型的级别。'model' 可以是下面字符串中的一个：

'interaction'——包含常数项、线性项核交互项。

'quadratic'——交互项加上平方项。

'purequadratic'——包含常数项、线性项和平方项。

regstats 函数提供了下面这些诊断统计量：

来自 QR 分解的 $\boldsymbol{Q}$ 矩阵（酉矩阵）；

来自 QR 分解的 $\boldsymbol{R}$ 矩阵（三角阵）；

诸系数；

诸系数的协方差；

诸拟合值；

诸残差；

均方误差；

中心化杠杆值；

Hat 矩阵；

剔除第 i 个变量以后得到的协方差；

剔除第 i 个变量以后得到的相关系数；

标准化残差；

学生化残差；

回归系数（β）的变化；

拟合值的变化；

拟合数据的比例变化；

协方差的变化；

Cook 距离。

算法：

一般的线性回归模型为

$$\boldsymbol{y}=\boldsymbol{X}\beta+\varepsilon$$

式中：$\boldsymbol{y}$ 为一 $n\times1$ 的响应向量，$\boldsymbol{X}$ 为一 $n\times p$ 的预测矩阵，β 为一 $p\times1$ 的参数向量，ε 为一 $n\times p$ 的随机干扰向量。

令 $\boldsymbol{X}=\boldsymbol{Q}*\boldsymbol{R}$，其中 $\boldsymbol{Q}$ 和 $\boldsymbol{R}$ 源于 $\boldsymbol{X}$ 的 QR 分解。这两个矩阵对于计算许多回归统计量都是很有用的。

进行 β 的最小二乘估计的标准文本公式为

$$\hat{\beta}=b=(X'X)^{-1}X'y$$

但该定义的数值属性比较差，特别是计算 $(X'X)^{-1}$ 时，既费时又不精确。

对于 β 而言，数值上稳定的 MATLAB 代码为：b = R\(Q'*y);

回归诊断图形用户界面如图 4-10 所示，用户可在图中进行参数选择。

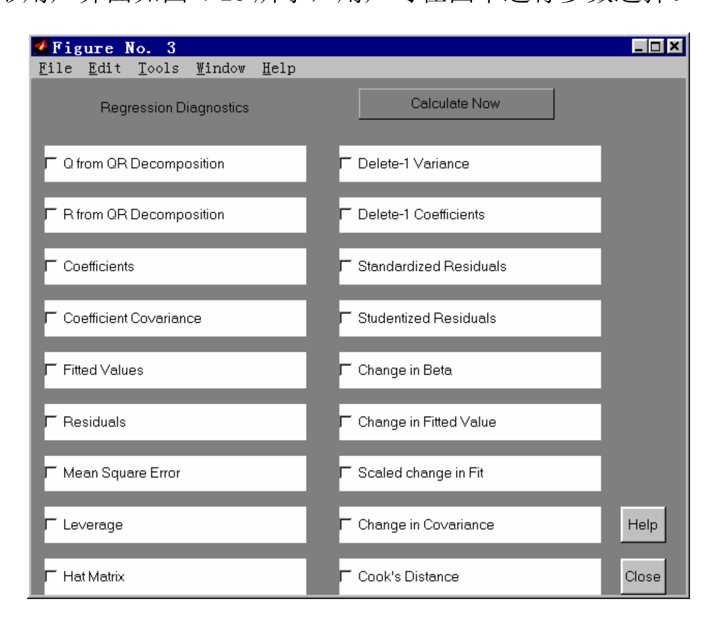


图 4-10　回归诊断图形用户界面

5．stepwise 函数

用 stepwise 函数进行逐步回归，其调用格式为：

● stepwise(X, y) 拟合 y 与 X 的列之间的回归模型。它显示 3 个图形窗口来交互地控制模型中项目的逐步引入和剔除。

● stepwise(X, y, inmodel) 允许控制初始回归模型的项目。值向量 inmodel 是矩阵 X 的列的待定系数，包含在初始模型中。

● stepwise(X, y, inmodel, alpha) 允许控制拟合系数的置信区间宽度。alpha 检验模型中每一项的显著性水平。默认时，alpha =1-(1-0.025)(1/p) ，其中 p 为 X 中的列数，它将为所有系数生成 95%置信区间。

最小二乘系数在图中用绿色充填的圆来表示。若某系数的置信区间穿过白色零线，则说明该系数与 0 没有显著差异。显著的模型项用实线表示，不显著的项用点线表示。

单击置信区间线来微调模型系数的状态。若置信区间线为绿色，则说明该项在模型中。若为红色，则说明该项不在模型中。

用弹出式菜单 Export 将变量剔除到基本工作空间。

4.2.3 应用实例

【例 4-12】 某种水泥在凝固时放出的热量(单位:卡/克)Y 与水泥中下列 4 种化学成分所占的百分比有关：

x1：$3CaO \cdot Al_2O_3$
x2：$3CaO \cdot SiO_2$
x3：$4CaO \cdot Al_2O_3 \cdot Fe_2O_3$
x4：$2CaO \cdot SiO_2$

现测得到 13 组数据，见表 4-6。要求建立热量与水泥化学成分之间的经验回归关系式。

表 4-6 热量–水泥化学成分数据

编号	x1	x2	x3	x4	y
1	7	26	6	60	78.5
2	1	29	15	52	74.3
3	11	56	8	20	104.3
4	11	31	8	47	87.6
5	7	52	6	33	95.9
6	11	55	9	22	109.2
7	3	71	17	6	102.7
8	1	31	22	44	72.5
9	2	54	18	22	93.1
10	21	47	4	26	115.9
11	1	40	23	34	83.8
12	11	66	9	12	113.3
13	10	68	8	12	109.4

%输入自变量和因变量。

```
X=[ 7    26     6    60
    1    29    15    52
   11    56     8    20
   11    31     8    47
    7    52     6    33
   11    55     9    22
    3    71    17     6
    1    31    22    44
    2    54    18    22
   21    47     4    26
    1    40    23    34
   11    66     9    12
   10    68     8    12];
y=[78.5 74.3 104.3 87.6 95.9 109.2 102.7 72.5 93.1 115.9 83.8 113.3 109.4]';
```

%进行一般多元回归分析。

```
[b,bint,r,rint,stats] = regress(y,X)
 b =
    2.1930
    1.1533
    0.7585
    0.4863
bint =
    1.7739   2.6122
    1.0449   1.2618
    0.3977   1.1194
    0.3926   0.5800
r =
   -0.5680
    1.9943
   -0.2042
   -1.2017
   -0.0239
    4.1180
   -1.5779
   -3.5314
    2.0821
   -0.0386
    1.4933
    0.3946
   -2.8605
rint =
   -4.7218   3.5858
   -2.6478   6.6363
   -5.6325   5.2240
   -6.1767   3.7733
```

```
   -4.7305   4.6828
   -0.2283   8.4642
   -6.0304   2.8747
   -7.1130   0.0502
   -2.8664   7.0306
   -3.3262   3.2490
   -2.9409   5.9276
   -4.8088   5.5980
   -7.3850   1.6639
stats =
    0.9860  152.6920    0.0000
```

上面的结果中，b 为自变量的系数向量，故全回归的回归结果为

```
y=2.1930X1+1.1533X2+0.7585X3+0.4863X4
```

bint 为各系数的置信区间。r 和 rint 为对应个案系数的残差和残差置信区间，可见各残差值均较小。stats 向量的值分别为相关系数的平方、F 值和显著性概率 p。相关系数平方值 R^2=0.9860，说明模型拟合程度相当高。显著性概率 p=0.0000，小于 0.05，故拒绝零假设，认为回归方程中至少有一个自变量的系数不为零，回归方程有意义。

%计算中心化杠杆值

```
h = leverage(X)
h =
    0.5503
    0.3332
    0.5769
    0.2952
    0.3576
    0.1242
    0.3671
    0.4085
    0.2943
    0.7004
    0.4255
    0.2630
    0.3037
```

%回归诊断

```
regstats(y,X)
```

输入上面的命令以后，打开如图 12-14 所示的 GUI 界面，选择所有的选项，并用默认变量名确定生成的统计量名称。本例的回归诊断结果如下：

```
>> Q                % Q 是源于 QR 分解的酉矩阵
Q =
    -0.2774   -0.0226    0.4168    0.4567   -0.3009
    -0.2774   -0.3171    0.2905    0.1791    0.1982
    -0.2774    0.1736   -0.1087    0.0572    0.6744
    -0.2774    0.1736    0.3677    0.0146    0.2297
    -0.2774   -0.0226   -0.0786    0.5010   -0.1515
```

```
    -0.2774    0.1736   -0.0897   -0.0244    0.0919
    -0.2774   -0.2189   -0.4868   -0.0545   -0.0481
    -0.2774   -0.3171    0.2524   -0.3767   -0.1595
    -0.2774   -0.2680   -0.1743   -0.0906    0.3270
    -0.2774    0.6644    0.1780   -0.3658   -0.1287
    -0.2774   -0.3171    0.0809   -0.4412   -0.2163
    -0.2774    0.1736   -0.2993   -0.0057   -0.2575
    -0.2774    0.1246   -0.3489    0.1504   -0.2587
```

```
>> R                %R 为源于 QR 分解的上三角阵
R =
   -3.6056  -26.9030  -173.6212   -42.4346  -108.1665
         0   20.3772    12.3215   -18.2859   -14.2316
         0         0   -52.4774    -1.1199    54.6073
         0         0          0   -12.5172    12.8688
         0         0          0          0    -3.4497
>> beta             %各自变量对应的回归系数
beta =
   62.4054
    1.5511
    0.5102
    0.1019
   -0.1441
>> covb             % 回归系数的协方差
covb =
  1.0e+003 *
    4.9099  -0.0505  -0.0506  -0.0517  -0.0496
   -0.0505   0.0006   0.0005   0.0006   0.0005
   -0.0506   0.0005   0.0005   0.0005   0.0005
   -0.0517   0.0006   0.0005   0.0006   0.0005
   -0.0496   0.0005   0.0005   0.0005   0.0005
>> yhat             % 响应数据的拟合值
yhat =
   78.4952
   72.7888
  105.9709
   89.3271
   95.6492
  105.2746
  104.1487
   75.6750
   91.7217
  115.6185
   81.8090
  112.3270
  111.6943
```

```
>> r                    % 残差
   r =
       0.0048
       1.5112
      -1.6709
      -1.7271
       0.2508
       3.9254
      -1.4487
      -3.1750
       1.3783
       0.2815
       1.9910
       0.9730
      -2.2943
>> mse                  % 均方误差
   mse =
       5.9830
>> leverage             % 中心化杠杆值
   leverage =
       0.5503
       0.3332
       0.5769
       0.2952
       0.3576
       0.1242
       0.3671
       0.4085
       0.2943
       0.7004
       0.4255
       0.2630
       0.3037
>> hatmat               % "Hat"矩阵
   hatmat =
     Columns 1 through 7
      0.5503    0.2274   -0.1491    0.1638    0.3191   -0.0032   -0.1314
      0.2274    0.3332    0.1341    0.1768    0.1210    0.0096   -0.0144
     -0.1491    0.1341    0.5769    0.2229    0.0080    0.1774    0.0563
      0.1638    0.1768    0.2229    0.2952    0.0166    0.0949   -0.1519
      0.3191    0.1210    0.0080    0.0166    0.3576    0.0539    0.1001
     -0.0032    0.0096    0.1774    0.0949    0.0539    0.1242    0.0795
     -0.1314   -0.0144    0.0563   -0.1519    0.1001    0.0795    0.3671
      0.0653    0.1517   -0.1347    0.0725   -0.1003   -0.0062    0.0517
     -0.1295    0.1598    0.2647    0.0401    0.0017    0.0783    0.2097
```

```
    0.0077   -0.1731    0.0652    0.2228   -0.1159    0.1734   -0.1291
   -0.0186    0.0791   -0.1581   -0.0045   -0.1105    0.0055    0.1414
    0.0231   -0.1171   -0.0343   -0.0622    0.1327    0.1104    0.1973
    0.0751   -0.0883   -0.0293   -0.0870    0.2161    0.1024    0.2237
  Columns 8 through 13
    0.0653   -0.1295    0.0077   -0.0186    0.0231    0.0751
    0.1517    0.1598   -0.1731    0.0791   -0.1171   -0.0883
   -0.1347    0.2647    0.0652   -0.1581   -0.0343   -0.0293
    0.0725    0.0401    0.2228   -0.0045   -0.0622   -0.0870
   -0.1003    0.0017   -0.1159   -0.1105    0.1327    0.2161
   -0.0062    0.0783    0.1734    0.0055    0.1104    0.1024
    0.0517    0.2097   -0.1291    0.1414    0.1973    0.2237
    0.4085    0.0999    0.0695    0.3986   -0.0105   -0.0660
    0.0999    0.2943   -0.1411    0.1171   -0.0011    0.0061
    0.0695   -0.1411    0.7004    0.0699    0.1742    0.0759
    0.3986    0.1171    0.0699    0.4255    0.0558   -0.0012
   -0.0105   -0.0011    0.1742    0.0558    0.2630    0.2687
   -0.0660    0.0061    0.0759   -0.0012    0.2687    0.3037
```

```
>> s2_i               % 剔除第 i 个值以后得到的方差
s2_i =
    6.8377
    6.3484
    5.8949
    6.2330
    6.8237
    4.3243
    6.3640
    4.4029
    6.4531
    6.7999
    5.8519
    6.6542
    5.7576
>> beta_i             % 剔除第 i 个值以后得到的相关系数
beta_i =
  Columns 1 through 7
   62.4851   48.0077  138.5862   79.3374   63.8150   51.0588   58.5170
    1.5505    1.7392    0.7874    1.4157    1.5489    1.6295    1.5548
    0.5093    0.6546   -0.2678    0.3268    0.4918    0.6243    0.5637
    0.1013    0.2682   -0.7099   -0.0687    0.0999    0.2159    0.1447
   -0.1450   -0.0139   -0.9162   -0.3073   -0.1612   -0.0247   -0.1122
  Columns 8 through 13
   37.2454   44.3291   66.3621   85.0849   72.4472   37.8708
    1.8797    1.7539    1.4583    1.2722    1.4448    1.7703
    0.7338    0.6926    0.4782    0.2967    0.4023    0.7846
```

```
   0.5187    0.2781    0.0384   -0.2437    0.0000    0.3164
   0.1042    0.0411   -0.1791   -0.3614   -0.2426    0.1030
```

>> standres　　　　　% 标准化残差

```
standres =
    0.0029
    0.7566
   -1.0503
   -0.8411
    0.1279
    1.7148
   -0.7445
   -1.6878
    0.6708
    0.2103
    1.0739
    0.4634
   -1.1241
```

>> studres　　　　　% 学生化残差

```
studres =
    0.0027
    0.7345
   -1.0581
   -0.8240
    0.1198
    2.0170
   -0.7218
   -1.9675
    0.6459
    0.1973
    1.0859
    0.4394
   -1.1459
```

>> dfbeta　　　　　% 回归系数的变化

```
dfbeta =
  Columns 1 through 7
  -0.0011    0.1995   -1.0953   -0.2367   -0.0188    0.1905    0.0538
   0.0008   -0.2451    1.0331    0.1781    0.0028   -0.1239   -0.0048
   0.0011   -0.1938    1.0828    0.2482    0.0238   -0.1854   -0.0717
   0.0007   -0.2139    1.0837    0.2215    0.0025   -0.1776   -0.0549
   0.0012   -0.1783    1.0970    0.2255    0.0226   -0.1980   -0.0436
  Columns 8 through 13
   0.4186    0.2484   -0.0530   -0.3273   -0.1359    0.3569
  -0.5144   -0.2622    0.1169    0.3787    0.1353   -0.3000
  -0.3601   -0.2427    0.0414    0.2982    0.1413   -0.3865
  -0.6437   -0.2248    0.0789    0.4630    0.1280   -0.2897
```

```
     -0.4081    -0.2514     0.0464     0.3099     0.1318    -0.3552
```

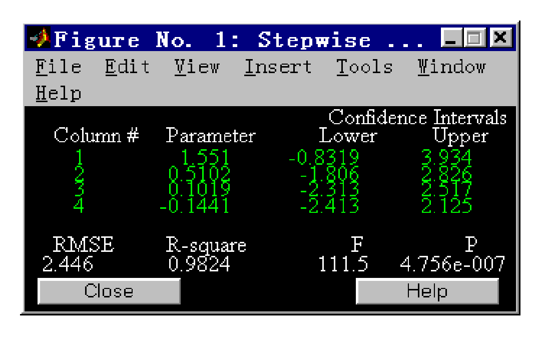
```
>> dffit                % 拟合值的变化
dffit =
    0.0058
    0.7553
   -2.2787
   -0.7235
    0.1396
    0.5565
   -0.8402
   -2.1931
    0.5748
    0.6582
    1.4747
    0.3472
   -1.0008
>> dffits               % 拟合数据的比例变化
dffits =
    0.0030
    0.5193
   -1.2356
   -0.5333
    0.0894
    0.7594
   -0.5497
   -1.6352
    0.4171
    0.3016
    0.9345
    0.2625
   -0.7568
>> covratio             % 方差的变化
covratio =
    1.1405
    1.1151
    2.5457
    1.1562
    0.8066
    5.7885
    1.1603
    7.8338
    0.9708
    1.7601
    1.9444
    0.7973
```

```
        1.7401
>> cookd                    % Cook 距离
    cookd =
        0.0000
        0.0572
        0.3009
        0.0593
        0.0018
        0.0834
        0.0643
        0.3935
        0.0375
        0.0207
        0.1708
        0.0153
        0.1102
```

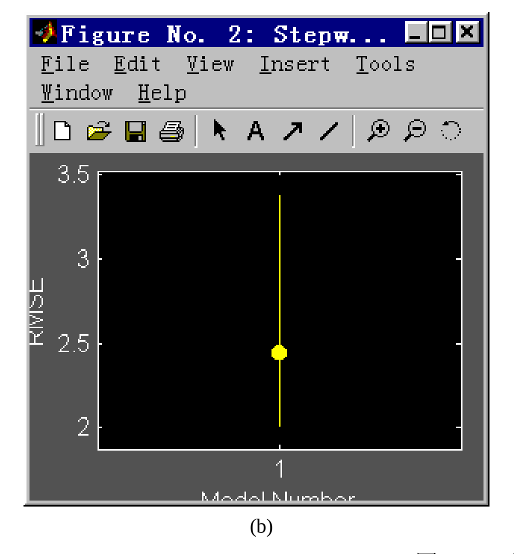
%逐步回归

```
    stepwise(X,y)
```

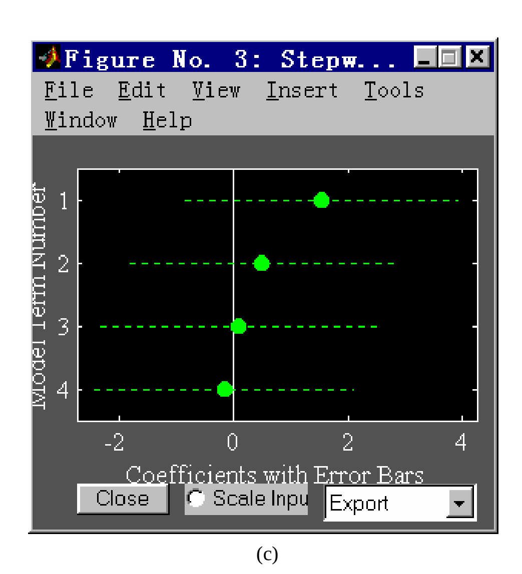
运行该命令，将打开 3 个图形窗口，如图 4-11 所示。



(a)



(b)



(c)

图 4-11　逐步回归分析图

图 4-11(a)所示窗口中包含各变量的回归系数及其 95%置信区间。另外还有均方差（RMSE）、回归系数的平方（R-square）、F 值(F)和显著性概率值（P）。

图 4-11(b)中为回归模型的均方差示意图。它用一个黄色填充的小圆圈表示对应模型的均方差。通过比较均方差，可以评价回归模型的优劣。

图 4-11(c)中用误差条图表示各变量的系数值。图中的填充圆圈表示对应变量的系数值，用两侧延伸的虚线表示对应系数值的置信区间。通过单击图中的圆圈或虚线，可以转换对应变量的引入或剔除状态。单击后，如果它们变成红色，则表示在当前模型中剔除对应变量；如果它们变成绿色，则表示在当前模型中引入对应变量。每次引入或剔除变量以后，前面两个图中都会发生相应的变化。选择图中的“Scale input”单选钮，将自变量的数据进行标准化以后再输入。利用“Export”下拉式列表框，可以确定有关信息的输出，包括回归系数、回归系数的置信区间、引入变量和剔除变量等。

默认时，系统将所有自变量引入到回归模型中，如上面模型中所示。与前面的全回归不同的是，逐步回归的模型中包含常数项。一般用下式计算常数项：

mean(y)-mean(X(:,in))*beta(in)

其中，in 为引入变量向量名，beta 为回归系数向量名，mean 表示求均值。

对于本例：

- 当不剔除变量时，回归模型为

  y=1.551X1+0.5102X2+0.1019X3-0.1441X4+62.4054

  回归系数平方值为 0.9824，均方差为 2.446。
- 当剔除变量 4 时，回归模型为

  y=1.696X1+0.6569X2+0.25X3+48.1936

  回归系数平方值为 0.9823，均方差为 2.312。
- 当剔除变量 3 和变量 4 时，回归模型为

  y=1.468X1+0.6623X2+52.5773

  回归系数平方值为 0.9787，均方差为 2.406。

进一步比较以后，可以发现，第 3 个模型具有较高的回归系数和较小的均方差，并且变量个数较少，所以是最合适的模型。

4.2.4 岭回归

1．基本数学原理

给定线性模型 $\boldsymbol{y}=\boldsymbol{X\beta}+\boldsymbol{\varepsilon}$，$\boldsymbol{\beta}$ 的岭估计为

$$\boldsymbol{b}=(\boldsymbol{X}'\boldsymbol{X}+k\boldsymbol{I})^{-1}\boldsymbol{X}'\boldsymbol{y}$$

式中：

$\boldsymbol{X}$ 为 $n\times p$ 的矩阵，$\boldsymbol{y}$ 为 $n\times 1$ 的观测量向量，k 为一标量常数（岭参数）。

当 $k=0$ 时， $\boldsymbol{b}$ 为最小二乘估计量。对于增加的 k，$\boldsymbol{b}$ 的偏倚增加，但 $\boldsymbol{b}$ 的方差减小。对于条件较差的 $\boldsymbol{X}$，方差的减小量比偏倚的弥补量要大。

岭回归分析中的关键问题是如何选择 k 值，适当选择 k 值可降低均方误差。

2．有关函数介绍

用 ridge 函数对岭回归进行参数估计。其调用格式为：b = ridge(y,X,k)，它返回岭回归系

数向量 **b**。

3．应用实例

【例 4-13】　本例显示当 *k* 值增加时系数是如何变化的。它使用了 hald 数据集。

```
load hald;
b = zeros(4,100);
kvec = 0.01:0.01:1;
count = 0;
for k = 0.01:0.01:1
count = count + 1;
b(:,count) = ridge(heat,ingredients,k);
end
plot(kvec',b'),xlabel('k'),ylabel('b','FontName','Symbol')
```

生成的图形如图 4-12 所示。

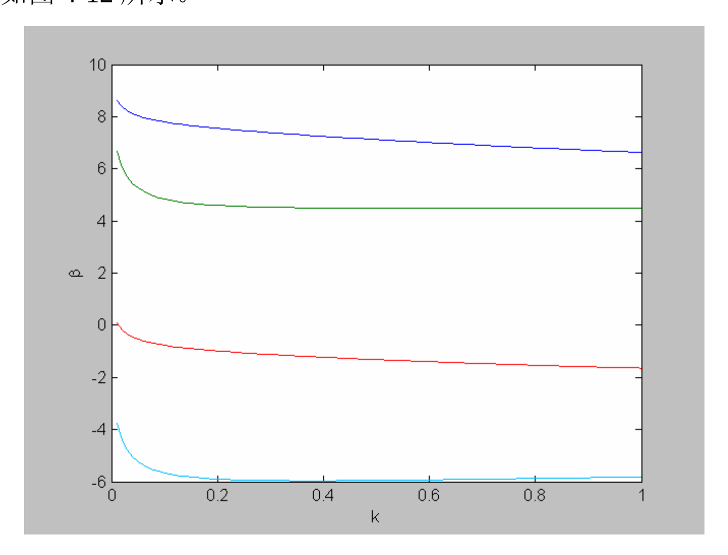


图 4-12　岭回归分析结果图示

4.3　扩展线性模型

用 glmfit 函数进行扩展线性模型拟合。

● b = glmfit(x, y, 'distr') 用响应 ***Y***。预测变量矩阵 ***X*** 和分布'distr'拟合广义线性模型。对于'distr'变量，可以选用下面的分布名称：'binomial', 'gamma', 'inverse gaussian', 'lognormal', 'normal'（默认选项）和'poisson'。

在大部分情况下，***Y*** 是响应观测值向量，但对于二项分布，***Y*** 为二列数组，第 1 列为测量次数，第 2 列为试验次数。***X*** 为矩阵，与 ***Y*** 有相同的行数，对应于每个观测值，包含预测变量的值。输出 **b** 为系数估计向量。

● b = glmfit(x, y, 'distr', 'link', 'estdisp', offset, pwts, 'const') 对拟合提供其他控制。'link'变量指定分布参数(μ)和预测变量(xb)组合之间的线性关系。在大多数情况下，'link'是表 4-7 字符串中的一个。也可以通过编写函数来定义自己的 link。例如，假设希望用 inline 函数定义一个倒数平方根连接，可以编写 mylink 变量并作为'link'变量的取值：

表 4-7　link 取值表

'link'	意　　义	默 认 设 置
'identity'	μ = xb	'normal'
'log'	log(μ) = xb	'poisson'
'logit'	log(μ / (1–μ)) = xb	'binomial'
'probit'	norminv(μ) = xb	
'comploglog'	log(-log(1–μ)) = xb	
'loglogl ink'	log(-log(μ)) = xb	
'reciprocal'	1/μ = xb	
'gamma'	p (a number) μp = xb	
'inverse gaussian' (p=-2)		

```
FL = inline('x.^-.5')
FD = inline('-.5*x.^-1.5')
FI = inline('x.^-2')
mylinks = {FL FI FD}
```

另外，也可以在各自的 M 文件中定义函数 FL、FD 和 FI，然后指定下面形式的 mylinks：

```
mylinks = {@FL @FD @FI}
```

estdisp 变量设置为 on，可以估计二项分布和泊松分布的分散参数。设置为 off。则给分散参数赋值 1.0。glmfit 函数总是为其他分布估计分散参数。

offset 参数和 pwts 参数可以是与 *Y* 长度相同的向量，也可以忽略（或指定为空向量）。offset 向量为一特殊的预测变量，其系数为 1.0。假设现在建立不同表面的缺陷个数模型，并希望创建一个模型，其中缺陷的个数期望与表面面积成比例。可以与泊松分布、对数连接函数和作为 offset 的对数表面面积一起，使用缺陷个数作为响应。

pwts 变量为优先权重向量。例如，如果响应值 *Y*(*i*)是 *f*(*i*)观测值的平均值，则可以使用 *f* 作为优先权重向量。

const 变量设置为 on(默认设置)，可以估计常数项，设置为 off，则忽略常数项。若希望得到常数项，则建议使用本变量。

● [b, dev, stats] = glmfit(...) 返回附加输出 dev 和 stats。dev 为解向量上的离差。该离差为残差平方和的推广。可以进行离差分析，比较多个模型、其他项的每一个子集，并检验项目较多的模型是否明显优于项目较少的模型。

stats 为含有下面域值的结构：

stats.dfe = 误差的自由度；

stats.s = 理论或估计的分散参数；

stats.sfit = 估计的分散参数；

stats.estdisp = 1（如果分散参数是估计的）或 0（如果分散参数是固定的）；

stats.beta = 系数估计向量（与 b 相同）；

stats.se = 系数估计的标准误差向量 b；

stats.coeffcorr = b 的相关矩阵；

stats.t = b 的 t 统计量；

stats.p = ***b*** 的 p 值；

stats.resid = 残差向量；

stats.residp = Pearson 残差向量；

stats.residd = 离差残差向量；

stats.resida = Anscombe 残差向量。

如果为二项分布或泊松分布估计分散参数，则 stats.s 等于 stats.sfit。同样，stats.se 的元素通过因子 stats.s 与它们的理论值相区别。

【例 4-14】 现有一批重量在 2100 磅至 4300 磅之间的小汽车。对于每个小汽车重量，有对应重量的汽车辆数，以及根据某些检验得到的认为性能较差的汽车的辆数。例如，21 辆小汽车中，有 8 辆重量为 3100 磅的汽车性能较差。

```
w = (2100:200:4300)';
poor = [1 2 0 3 8 8 14 17 19 15 17 21]';
total = [48 42 31 34 31 21 23 23 21 16 17 21]';
```

可以对这些数据用一些模型进行比较。首先，用 logit 模型和 probit 模型进行比较：

```
[bl,dl,sl] = glmfit(w,[poor total],'binomial');
[bp,dp,sp] = glmfit(w,[poor total],'binomial','probit');
dl
dl =
    6.4842
dp
dp =
    7.5693
```

logit 模型的偏差小于 probit 模型的，尽管该检验不是很正规，但可以看出选用 logit 模型进行拟合更合适。

可以通过比较两个 logit 模型来进行正规检验。我们已经用 ***w*** 作为线性预测量拟合了一个模型。现在用 ***w*** 中的线性项和平方项拟合另一个 logit 模型。如果对于平方项，没有真实的效应，则与具有某个自由度的卡方分布相比，它们偏差的差异将会较小。如：

```
[b2,d2,s2] = glmfit([w w.^2],[poor total],'binomial')
b2 =
   -7.3109
    0.0002
    0.0000
d2 =
    5.7815
s2 =
         beta: [3x1 double]
          dfe: 9
         sfit: 0.6556
      estdisp: 0
            s: 1
           se: [3x1 double]
    coeffcorr: [3x3 double]
            t: [3x1 double]
            p: [3x1 double]
        resid: [12x1 double]
```

```
        residp: [12x1 double]
        residd: [12x1 double]
        resida: [12x1 double]
dl-d2
ans =
   0.7027
chi2cdf(dl-d2,1)
ans =
   0.5981
```

对于具有某个自由度的卡方分布而言，0.7072 的差异不是偶然的，所以平方模型的拟合效果并不比线性模型的好多少。

下面是系数估计及它们的标准误差、t 统计量和线性模型的 *p* 值：

```
[b sl.se sl.t sl.p]
ans =
  -13.3801    1.3940   -9.5986    0.0000
    0.0042    0.0004    9.4474    0.0000
```

这表示我们不能更多地简化模型。交互项和斜率系数都与 0 有显著差异。

4.4　多项式拟合

（1）用 polyconf 函数进行多项式评价和置信区间估计。该函数的调用格式为：

● [Y, DELTA] = polyconf(p, X, S) 使用 polyfit 函数的选项输出 S 给出 Y 的 95%置信区间 Y ± DELTA。它假设 polyfit 函数数据输入的误差是独立正态的，并且方差为常数。

● [Y, DELTA] = polyconf(p, X, S, alpha) 给出 100(1-alpha)%置信区间。例如，alpha = 0.1 时生成 90%置信区间。若 p 为向量，它们的元素为幂次降序的多项式系数，就象 polyfit 函数的输出一样，polyconf(p, X)为 X 处多项式的值。若 X 为矩阵或向量，则在每个元素处计算多项式的值。

【例 4-15】 本例给出预测值和 90%置信区间，以计算平方矩阵第 100 列至 200 列 LU 因子分解的时间。

```
n = [100 100:20:200];
for i = n
A = rand(i,i);
tic
B = lu(A);
  t(ceil((i-80)/20)) = toc;
end
[p,S] = polyfit(n(2:7),t,3);
[time,delta_t] = polyconf(p,n(2:7),S,0.1)
time =
   0.0829    0.1476    0.2277    0.3375    0.4912    0.7032
delta_t =
   0.0064    0.0057    0.0055    0.0055    0.0057    0. 0064
```

（2）用 polyfit 函数进行多项式曲线拟合。该函数的调用格式为：

● p = polyfit(x, y, n) 计算数据(*x*, *y*)*n* 次多项式的最小二乘意义上的系数估计值，结果 *p*

为一行向量，其长度为 n+1，包含幂次降序的多项式系数。

$$p(x)=p_1x^n+p_2x^{n-1}+\ldots p_nx+p_{n+1}$$

● [p, S] = polyfit(x, y, n) 返回多项式系数向量 p 和矩阵 S。S 与 polyval 函数一起用，可以生成预测值的误差估计。若数据 ***y*** 的误差为独立正态，且方差为常数，则 polyval 函数将生成误差范围，其中包含至少 50%的预测值。

若不想用 polyval 函数或 polyconf 函数进行误差估计，可以忽略 S。

【例 4-16】

```
[p,S] = polyfit(1:10,[1:10] + normrnd(0,1,1,10),1)
p =
    1.0300    0.4561
S =
  -19.6214   -2.8031
         0   -1.4639
    8.0000         0
    2.3180         0
```

（3）用 polyval 函数进行多项式评价，其调用格式为：

● Y = polyval(p, X) 返回在值 *X* 处给定系数 p 的多项式的预测值。

● [Y, DELTA] = polyval(p, X, S) 使用选项输出 S。S 由 polyfit 函数生成，得到误差估计 Y+/-DELTA。若数据输入的误差是独立正态的，方差为常数，则 Y+/-DELTA 将至少包含 50%的预测值。

若 p 为一向量，它的元素为降序幂次多项式的系数，则 polyval(p, X)为 X 处多项式的值。若 X 为一矩阵或向量，则多项式在每一个元素处进行评估。

【例 4-17】 模拟函数 $y = x$, 添加标准离差为 0.1 的正态随机误差。然后使用 polyfit 来估计多项式的系数。

```
[p,S] = polyfit(1:10,(1:10) + normrnd(0,0.1,1,10),1);
X = magic(3);
[Y,D] = polyval(p,X,S)
Y =
    8.0696    1.0486    6.0636
    3.0546    5.0606    7.0666
    4.0576    9.0726    2.0516
D =
    0.0889    0.0951    0.0861
    0.0889    0.0861    0.0870
    0.0870    0.0916    0.0916
```

4.5 稳健回归

稳健回归，是指该回归方法相对于其他回归方法而言，受异常值的影响较小。

MATLAB 中用 robust 函数进行稳健回归。

● b = robustfit(X, Y) 用稳健回归来拟合，即将 Y 表达为 X 的列数据的函数并返回系数估计向量 b。robustfit 函数使用交互式的最小二乘算法，每一次迭代的权重用 bisquare 函数和上一次迭代的残差一起进行计算。本算法将较小的权重赋给拟合较差的点。与一般的最小二乘

回归相比，本方法受异常值的影响较小。

- [b, stats] = robustfit(X, Y) 还返回一个 stats 结构，结构取值可以是下面域值中的一个：

stats.ols_s——源于最小二乘拟合的σ 估计（rmse）；

stats.robust_s——σ 的稳健估计；

stats.mad_s——用源于中值的残差的中值绝对差计算σ 估计，以在迭代拟合的过程中度量残差；

stats.s——σ 的最终估计，是 robust_*s* 和加权 ols_*s* 的均值和 robust_*s* 之间的大者；

stats.se——系数估计的标准误差；

stats.t——*b* 与 stats.se 之间的比率；

stats.p–stats.t 的 *p* 值；

stats.coeffcorr——系数估计的相关性估计；

stats.w——稳健拟合的权重向量；

stats.h——最小二乘拟合的中心化杠杆值向量；

stats.dfe——误差自由度；

stats.R——***X*** 矩阵 QR 分解中的 *R* 因子。

- robustfit 函数求系数估计的方差-协方差估计，即，V = inv(X'*X)*stats.s^2。标准误差和相关系数由 V 导出。
- [b, stats] = robustfit(X, Y, 'wfun', tune, 'const') 指定一个加权函数、一个调协常数和是否显示常数项。加权函数'wfun'可以是表 4-8 中的一个。用户也可以编写自己的加权函数。

表 4-8　加权函数表

加 权 函 数	意　　义	调 协 常 数
'andrews'	w = (abs(r)<pi) .* sin(r) ./ r	1.339
'bisquare'	w = (abs(r)<1) .* (1 – r.^2).^2	4.685
'cauchy'	w = 1 ./ (1 + r.^2)	2.385
'fair'	w = 1 ./ (1 + abs(r))	1.400
'huber'	w = 1 ./ max(1, abs(r))	1.345
'logistic'	w = tanh(r) ./ r	1.205
'talwar'	w = 1 * (abs(r)<1)	2.795
'welsch'	w = exp(-(r.^2))	2.985

加权函数中的 r 的计算表达式为

resid/(tune*s*sqrt(1-h))

式中，resid 为源于前一次迭代的残差向量，tune 为调协常数，h 为源于最小二乘拟合的中心化杠杆值向量，s 为误差项的标准离差估计，且有

s = MAD/0.6745

数值 MAD 为至中值的残差的中值绝对差。常数 0.6745 使得估计对于正态分布而言是无偏的。若 ***X*** 矩阵中有 *p* 列（包含常数项），则计算其中值时剔除最小的 p–1 个绝对差。

除了上面列出的函数名以外，'wfun'可以是'ols'，用于进行不加权一般最小二乘计算。

变量 tune 的值代替表中默认的调协常数。'const'的值可以是'on'（默认选项），此时添加一个常数项；也可以是 off'，忽略常数项。如果希望模型中包含常数项，将参数'const'设置为'on'

比在 ***X*** 矩阵中添加一个单位列（列中的值均为 1）好。

【例 4-18】 本例演示一个错误的点数据如何影响最小二乘拟合和稳健拟合。首先利用函数 y = 10–2*x 加上一些随机项生成简单数据集。然后改变一个 y 值来模拟一个异常值（它可能是一次错误的测量）：

```
x = (1:10)';
y = 10 - 2*x + randn(10,1);
y(10) = 0;
```

我们使用一般最小二乘法和稳健拟合来估计直线拟合方程。

```
bls = regress(y,[ones(10,1) x])
bls =
    8.6305
   -1.4721
brob = robustfit(x,y)
brob =
   10.5089
   -1.9844
```

带拟合线的散点图显示稳健拟合（实线）对数据的拟合程度最好，但忽略了异常值。最小二乘拟合（虚线）则受到异常值的影响，向异常值偏过去。

```
scatter(x,y)
hold on
plot(x,bls(1)+bls(2)*x,'g:')
plot(x,brob(1)+brob(2)*x,'r-')
```

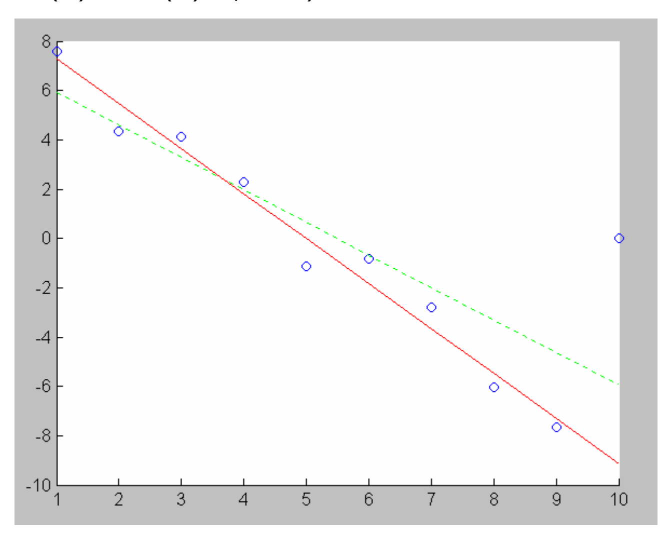


图 4-13 最小二乘拟合与稳健回归拟合的比较

4.6 二次响应面模型

用 rstool 函数进行交互式拟合及响应面可视化，其调用格式为：

- rstool(x, y) 显示 95%置信区间的交互预测图。该图的数据源于（x，y）数据的多元线性回归结果。

● rstool(x, y, 'model') 允许控制初始回归模型。“model”可以是下面字符串中的一个：

'interaction'——包含常数项、线性项和交互项；

'quadratic'——交互项加上平方项；

'purequadratic'——包含常数项、线性项和平方项。

● rstool(x, y, 'model', alpha) 用两条红色曲线来表示 100(1-alpha)%置信区间。例如，当 alpha = 0.01 时，给出 99%置信区间。

rstool 显示一个图形"向量"，其中的每一个图形对应于输入矩阵 x 的每一列。响应变量 y 为一列向量，与 **x** 的行数相匹配。

● rstool(x, y, 'model', alpha, 'xname', 'yname') 在图中用字符串矩阵'xname'来标注 x 轴，用‘yname’来标注 y 轴。它适用于所有的图形。

在图中拖拉白色带点参考线，同时观察预测值的更新情况。可以通过在编辑文本域中输入 x 值来获得指定的预测值。使用标签为“Model”的弹出式菜单可以交互地改变模型。使用标签为“Export”的弹出式菜单可以将指定变量移到基本工作空间。

【例 4-19】 一只红铃虫的产卵数与温度有关，下面是有关的数据：

温　度	21	23	25	27	29	32	35
产卵数	7	11	21	24	66	115	325

试找出一种较佳的经验回归函数。

```
x=[21  23  25  27  29  32  35]';
y=[7  11  21  24  66  115  325]';
rstool(x,y)
```

生成如图 4-14 所示的拟合图（默认显示线性拟合图）。在 resports 下拉式列表框中选择 All 选项，在命令窗口输入下面的命令，得到拟合函数的系数值、均方差和残差。

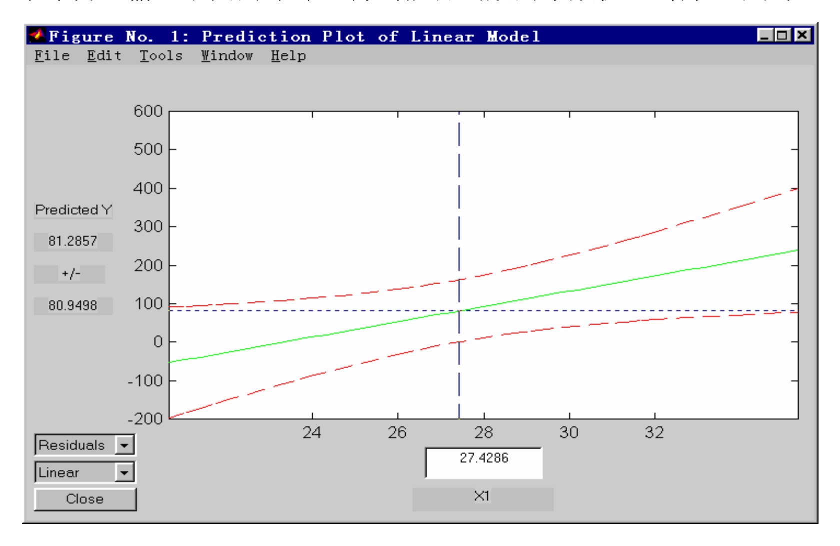


图 4-14　线性拟合图

```
>>beta
beta =
 -463.7311
   19.8704
>>rmse
```

```
rmse =
   62.9586
>>residuals
residuals =
   53.4526
   17.7118
  -12.0290
  -48.7698
  -46.5106
  -57.1219
   93.2669
```

故，线性函数为

产卵数=–463.7311+19.8704*温度

用该函数进行拟合的均方差为 62.9586。residuals 中列出了各测量值的残差。

在 linear 下拉式列表框中选择 Quadratic 选项，生成如图 4-15 所示的拟合图。在 Reports 下拉式列表框中选择 All 选项，在命令窗口输入下面的命令，得到拟合函数的系数值、均方差和残差。

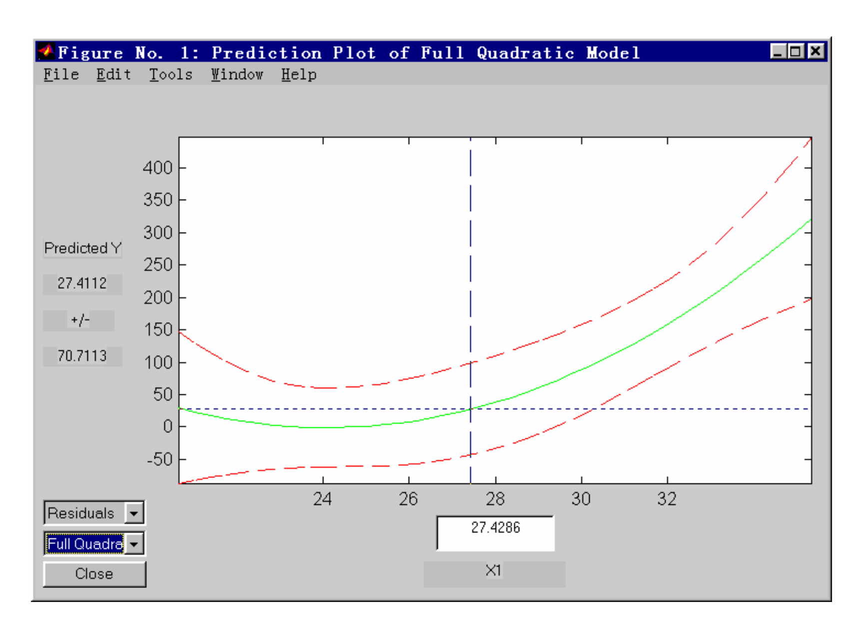


图 4-15　二次拟合图

```
>> beta
beta =
    1.0e+003*
    1.4821
   -0.1231
    0.0026
>> rmse
rmse =
   28.2266
>> residuals
residuals =
```

```
-16.6846
  8.7687
 19.7977
  3.4022
  5.5824
-43.4430
 22.5766
```

从上面的结果可以看出，采用下面的二次函数

产卵数=1482.1–123.1*温度+2.6*温度^2

可以显著减小均方差（现在均方差为 28.2266），拟合残差也显著减小，故该拟合比线性拟合更佳。

第 5 章　非线性模型

5.1　非线性最小二乘

在科学数据的处理过程中，常常需要建立变量之间的数学模型。这些数学模型往往是非线性的。非线性模型比线性模型更难拟合，通常是给定数学模型中待定参数的初始值以后，采用最小二乘迭代的方法进行求解。常用的方法有高斯-牛顿法、马夸尔特法等。

MATLAB 中提供了一系列函数进行非线性最小二乘计算，包括：nlinfit，nlintool，nlpredci，nlparci 和 nnls 等。

1．nlinfit 函数

用 nlinfit 函数进行非线性最小二乘数据拟合。该函数使用高斯-牛顿算法，调用格式为：

- beta = nlinfit(X, y, 'model', beta0)　返回‘model’中描述的非线性函数的系数。'model'为一自定义函数，有以下形式：$\hat{\boldsymbol{y}} = \boldsymbol{f}(\beta, \boldsymbol{X})$。返回 $\boldsymbol{y}$ 的预测值（$\boldsymbol{y}$ 给定初值）和独立变量 $\boldsymbol{X}$。beta0 为系数的初值向量。矩阵 $\boldsymbol{X}$ 有多列，每一列为一变量。响应值 $\boldsymbol{y}$ 为一列向量，与 $\boldsymbol{X}$ 有相同的行数。
- [beta, r, J] = nlinfit(X, y, 'model', beta0)　返回拟合系数（beta）、残差（r）和雅可比矩阵 J，用于 nlintool 函数，生成预测值的误差估计。

【例 5-1】　在化工生产中获得的氯气的等级 y 随生产时间 x 下降。假定在 $x \geqslant 8$ 时，y 与 x 之间有如下形式的非线性模型：

$$y = a + (0.49 - a)\mathrm{e}^{-b(x-8)}$$

现收集了 44 组数据：

x	y	x	y	x	y
8	0.49	16	0.43	28	0.41
8	0.49	18	0.46	28	0.40
10	0.48	18	0.45	30	0.40
10	0.47	20	0.42	30	0.40
10	0.48	20	0.42	30	0.38
10	0.47	20	0.43	32	0.41
12	0.46	20	0.41	32	0.40
12	0.46	22	0.41	34	0.40
12	0.45	22	0.40	36	0.41
12	0.43	24	0.42	36	0.38
14	0.45	24	0.40	38	0.40
14	0.43	24	0.40	38	0.40
14	0.43	26	0.41	40	0.39
16	0.44	26	0.40	42	0.39
16	0.43	26	0.41		

要求利用该数据求 a，b 的值，以确定模型。

首先定义非线性函数的 M 文件 sta67_1m.m：

```
function yy=model(beta0,x)
  a=beta0(1);
```

```
    b=beta0(2);
    yy=a+(0.49-a)*exp(-b*(x-8));
```

然后在命令窗口中输入以下命令行：

```
x=[ 8.00  8.00 10.00 10.00 10.00 10.00 12.00 12.00 12.00 14.00 14.00 14.00…
   16.00 16.00 16.00 18.00 18.00 20.00 20.00 20.00 20.00 22.00 22.00 24.00…
   24.00 24.00 26.00 26.00 26.00 28.00 28.00 30.00 30.00 30.00 32.00 32.00…
   34.00 36.00 36.00 38.00 38.00 40.00 42.00]';
y=[0.49 0.49 0.48 0.47 0.48 0.47 0.46 0.46 0.45 0.43 0.45 0.43 0.43 0.44 0.43…
   0.43 0.46 0.42 0.42 0.43 0.41 0.41 0.40 0.42 0.40 0.40 0.41 0.40 0.41 0.41…
   0.40 0.40 0.40 0.38 0.41 0.40 0.40 0.41 0.38 0.40 0.40 0.39 0.39]';
beta0=[0.30 0.02];
betafit = nlinfit(x,y,'sta67_1m',beta0)
betafit =
    0.3896
    0.1011
```

故模型中 a=0.3896，b=0.1011，最终模型为

$$y = 0.3896 + (0.49 - 0.3896)\mathrm{e}^{-0.1011(x-8)}$$

【例 5-2】 做财政收入预测。财政收入与国民收入、工农业总产值、人口、就业人口、固定资产投资等因素有关。采用岭回归分析构造预测模型。以财政收入作为因变量 y，自变量为国民收入(x_1)、工业总产值(x_2)、农业总产值(x_3)、总人口(x_4)、就业人口(x_5)、固定资产投资(x_6)。原始数据见表 5-1。其样本数 n=30，自变量数 p=6。

表 5-1　原始数据表

年　份	国民收入（亿元）	工业总产值（亿元）	农业总产值（亿元）	总人口（万人）	就业人口（万人）	固定资产投资（亿元）	财政收入（亿元）
1952	598	349	461	57482	20729	44	184
1953	586	455	475	58796	21364	89	216
1954	707	520	491	60266	21832	97	248
1955	737	558	529	61465	22328	98	254
1956	825	715	556	62828	23018	150	268
1957	837	798	575	64653	23711	139	286
1958	1028	1235	598	65994	26600	256	357
1959	1114	1681	509	67207	26173	338	444
1960	1079	1870	444	66207	25880	380	506
1961	757	1156	434	65859	25590	138	271
1962	677	964	461	67295	25110	66	230
1963	779	1046	514	69172	26640	85	266
1964	943	1250	584	70499	27736	129	323
1965	1152	1581	632	72538	28670	175	393
1966	1322	1911	687	74542	29805	212	466
1967	1249	1647	697	76368	30814	156	352
1968	1187	1565	680	78534	31915	127	303
1969	1372	2101	688	80671	33225	207	447
1970	1638	2747	767	82992	34432	312	564

续表

年 份	国民收入（亿元）	工业总产值（亿元）	农业总产值（亿元）	总人口（万人）	就业人口（万人）	固定资产投资（亿元）	财政收入（亿元）
1971	1780	3156	790	85229	35620	355	638
1972	1833	3365	789	87177	35854	354	658
1973	1978	3684	855	89211	36652	374	691
1974	1993	3696	891	90859	37369	393	655
1975	2121	4254	932	92421	38168	462	692
1976	2052	4309	955	93717	38834	443	657
1977	2189	4925	971	94974	39377	454	723
1978	2475	5590	1058	96259	39856	550	922
1979	2702	6065	1150	97542	40581	564	890
1980	2791	6592	1194	98705	41896	568	826
1981	2927	6862	1273	100072	43280	496	810

首先编制非线性函数 M 文件 sta67_2m.m：

```
function yy=model(beta0,X)
  a=beta0(1);
  b=beta0(2);
  c=beta0(3);
  d=beta0(4);
  e=beta0(5);
  f=beta0(6);
  x1=X(:,1);
  x2=X(:,2);
  x3=X(:,3);
  x4=X(:,4);
  x5=X(:,5);
  x6=X(:,6);
  yy=a*x1+b*x2+c*x3+d*x4+e*x5+f*x6;
```

然后在命令窗口输入以下命令行：

```
X=[598.00    349.00     461.00  57482.00   20729.00     44.00
    586.00    455.00     475.00   58796.00   21364.00     89.00
    707.00    520.00     491.00   60266.00   21832.00     97.00
    737.00    558.00     529.00   61465.00   22328.00     98.00
    825.00    715.00     556.00   62828.00   23018.00    150.00
    837.00    798.00     575.00   64653.00   23711.00    139.00
   1028.00   1235.00     598.00   65994.00   26600.00    256.00
   1114.00   1681.00     509.00   67207.00   26173.00    338.00
   1079.00   1870.00     444.00   66207.00   25880.00    380.00
    757.00   1156.00     434.00   65859.00   25590.00    138.00
    677.00    964.00     461.00   67295.00   25110.00     66.00
    779.00   1046.00     514.00   69172.00   26640.00     85.00
```

```
  943.00   1250.00    584.00   70499.00   27736.00   129.00
 1152.00   1581.00    632.00   72538.00   28670.00   175.00
 1322.00   1911.00    687.00   74542.00   29805.00   212.00
 1249.00   1647.00    697.00   76368.00   30814.00   156.00
 1187.00   1565.00    680.00   78534.00   31915.00   127.00
 1372.00   2101.00    688.00   80671.00   33225.00   207.00
 1638.00   2747.00    767.00   82992.00   34432.00   312.00
 1780.00   3156.00    790.00   85229.00   35620.00   355.00
 1833.00   3365.00    789.00   87177.00   35854.00   354.00
 1978.00   3684.00    855.00   89211.00   36652.00   374.00
 1993.00   3696.00    891.00   90859.00   37369.00   393.00
 2121.00   4254.00    932.00   92421.00   38168.00   462.00
 2052.00   4309.00    955.00   93717.00   38834.00   443.00
 2189.00   4925.00    971.00   94974.00   39377.00   454.00
 2475.00   5590.00   1058.00   96259.00   39856.00   550.00
 2702.00   6065.00   1150.00   97542.00   40581.00   564.00
 2791.00   6592.00   1194.00   98705.00   41896.00   568.00
 2927.00   6862.00   1273.00   100072.0   43280.00   496.00];
y=[184.00 216.00 248.00 254.00 268.00 286.00 357.00 444.00 506.00 …
   271.00 230.00 266.00 323.00 393.00 466.00 352.00 303.00 447.00 …
   564.00 638.00 658.00 691.00 655.00 692.00 657.00 723.00 922.00 …
   890.00 826.00 810.0]';
beta0=[0.50 -0.03 –0.60 0.01 –0.02 0.35];
betafit = nlinfit(X,y,’sta67_2m.m’,beta0)
betafit =
        0.5243
       -0.0294
       -0.6304
        0.0112
       -0.0230
        0.3658
```

故财政收入与各因素之间有下面的关系：

财政收入=0.5243*国民收入– 0.0294*工业总产值– 0.6304*农业总产值

+ 0.0112*总人口– 0.0230*就业人口+ 0.3658*固定资产投资

2．nlintool 函数

使用 nlintool 函数可对数据进行非线性方程拟合并用交互图形显示，其调用格式为：

● nlintool(x, y, 'model', beta0) 为一预测图，它提供数据(x, y)的非线性曲线拟合。它用两条红色曲线来表示预测值的 95%置信区间。beta0 为一向量，包含参数的初值。

● nlintool(x, y, 'model', beta0, alpha) 用图形表示预测值的 100(1 - alpha)%置信区间。当 x 为矩阵时，将为每一列单独生成图形，应变量 y 为一列变量，对应于 x 的行。Alpha（α）的默认值为 0.05，生成 95%置信区间。

● nlintool(x,y,'model',beta0,alpha,'xname','yname') 用字符串矩阵'xname'和'yname'来标注图形的 x 变量和 y 变量。

【例 5-3】 现评价坝肩岩体质量。对部分平硐进行 RMR 分级和 Q 分级以后，得到一系列的 RMR 值和 Q 值。

```
RMR  30  34  31  37  46  47  48  46  46  49  50  50  60 58  60 56 62 69 70 72 72
Q     1  1.5  2  1.8  5  4.5  4  5  5.5 5.5  7  7.5 10 9.5 12 14 20 24 24 27 30
RMR  75  76  72  73  78  76  80  80  78  78  81  81  78  86  88  88
Q    29  30  31 31  32  34  73  75  40  39  41  45  46  49  57  65
```

假设 RMR 和 Q 之间满足下面的关系式，试确定其中的待定系数。

$$RMR = e^{(a+b*\log(Q))}$$

首先编制M文件sta67_3m.m，确定非线性模型：

```
function yy=model(beta0,X)
    a=beta0(1);
    b=beta0(2);
    yy=exp(a+b*log(X));
```

然后在命令窗口中输入下面的命令行：

```
X=[30  34 31 37 46 47 48 46 46 49 50 50 60 58 60 56 62 69 70 72 72…
   75  76 72 73 78 76 80 80 78 78 81 81 78 86 88 88];
y=[1 1.5 2 1.8 5 4.5 4 5 5.5 5.5 7 7.5 10 9.5 12 14 20 24 24 27 30…
   29 30 31 31 32 34 73 75 40 39 41 45 46 49 57 65];
beta0=[3 0.2];      %赋初值
nlintool(X,y,'sta67_3m',beta0)
```

上面的命令行生成图 5-1，图中绿色曲线为拟合线，红色虚线为拟合的误差区间上界和下界线。

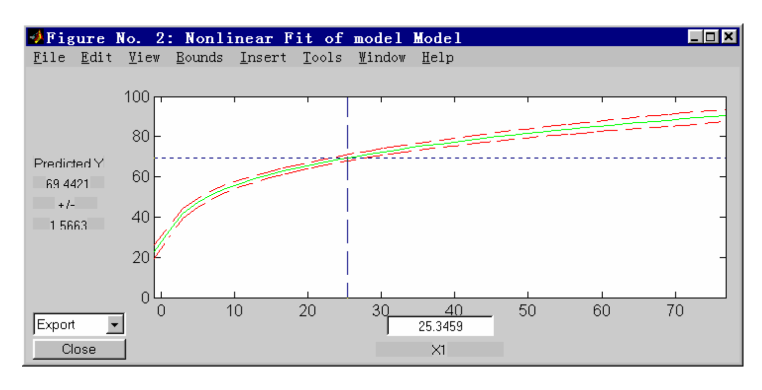


图 5-1 非线性拟合图形

```
>> beta
beta =
    3.4711
    0.2380
```

故非线性回归模型中的系数 a 和系数 b 分别为 3.4711 和 0.2380。最终确定的模型为：

$$RMR = e^{(3.4711+0.2380*\log(Q))}$$

```
>> betaci
betaci =
    3.4051    3.5371
    0.2185    0.2575
```

```
>> rmse
rmse =
    3.5353
>> residuals
residuals =
   -2.1721
   -1.4316
   -6.9426
   -0.0030
   -1.1890
    0.9796
    3.2518
   -1.1890
   -2.2717
    0.7283
   -1.1235
   -1.9699
    4.3470
    3.0223
    1.8788
   -4.2933
   -3.6352
    0.4539
    1.4539
    1.5051
   -0.2850
    3.2959
    3.7150
   -0.8513
    0.1487
    4.5961
    1.5292
   -9.3239
   -9.9004
    0.5922
    1.0572
    3.1359
    1.3915
   -2.0261
    4.7615
    3.7841
    1.1100
```

betaci，rmse 和 residuals 分别为系数的 95%置信区间、均方差和残差。

【例 5-4】 对于多变量的情况，沿用 nlinfit 函数例 5-2 的数据，在命令窗口中输入下面的命令行：

```
X=[……];              % 与 nlinfit 函数例 5-2 中的一样
y=[……];              % 与 nlinfit 函数例 5-2 中的一样
```

```
nlintool(X,y,'sta67_2m',beta0)
```

生成的非线性拟合模型如图 5-2 所示。

在 Output 下拉式列表框中选择 Parameters 选项，并确定向量名称为默认的 beta，输出拟合系数向量。

```
>>beta
beta =
    0.5243
   -0.0294
   -0.6304
    0.0112
   -0.0230
    0.3658
```

可见与前面的结果一致。

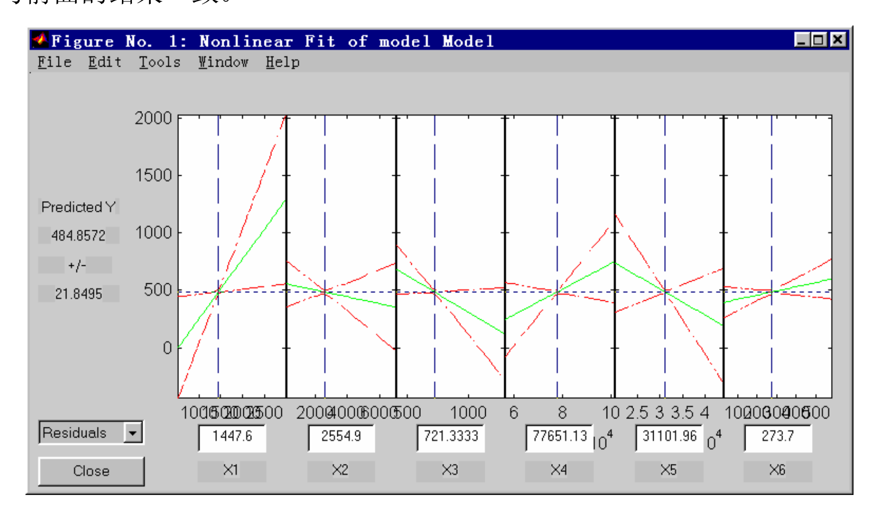


图 5-2　多变量的非线性拟合

3．nlpredci 函数

可利用 nlpredci 函数计算非线性模型预测值的置信区间，其调用格式如下：

● ypred = nlpredci(FUN, inputs, beta, r, J) 给定拟合参数（beta）、残差（r）和雅可比矩阵（J），返回预测响应。输入是非线性函数中独立变量的数值矩阵。

● [ypred, delta] = nlpredci(FUN, inputs, beta, r, J)还返回参数 delta，用于计算非线性最小二乘预测的置信区间。对于 r 的长度大于 beta 的长度，且 J 为列满秩矩阵的系统，置信区间的计算是有效的。区间[ypred-delta, ypred+delta]为指定输入值处函数真值的 95%异步置信区间。

● ypred = nlpredci(FUN, inputs, beta, r, J, alpha, 'simopt', 'predopt') 控制置信区间的类型。置信水平为 100(1-alpha)%。对于同步区间，'simopt'设为'on'。对于异步区间，则设为'off'(默认选项)。对于输入处的函数置信区间，'predopt'设为'curve'(默认选项)。对于新的响应值，则设为'observation'。

nlpredci 函数用 nlinfit 函数的输出作为输入。

【例 5-5】　继续使用 nlinfit 函数的实例，确定[100 300 80]处的预测函数值和它的置信区间的一半宽度。

```
load reaction
[beta,resids,J] = nlinfit(reactants,rate,@hougen,beta);
[ypred,delta] = nlpredci(@hougen,[100 300 80],beta,resids,J)
ypred =
       13
delta =
       1.4277
```

4．nlparci 函数

利用 nlparci 函数计算非线性模型中参数估计值的置信区间。

● nlparci(beta, r, J) 给定拟合参数（beta）、残差（r）和解处的雅可比矩阵（J），返回非线性最小二乘参数估计（beta）的 95%置信区间 ci。当（J）的行数大于（beta）的长度时，置信区间的计算是有效的。

nlparci 函数使用 nlinfit 函数的输出作为输入。

【例 5-6】 继续使用 nlinfit 函数的例子：

```
load reaction
[beta,resids,J] = nlinfit(reactants,rate,'hougen',beta);
ci = nlparci(beta,resids,J)
ci =
   -1.0798   3.3445
   -0.0524   0.1689
   -0.0437   0.1145
   -0.0891   0.2941
   -1.1719   3.7321
```

5．nnls 函数

可利用 nnls 函数进行非负最小二乘计算，其调用格式如下：

● x = nnls(A, b) 在最小二乘意义上求解方程组 $Ax = b$，约束条件为解向量包含非负元素: $x_j > 0, j = 1, 2, ..., n$。

● x = nnls(A, b, tol) 求解方程组并指定容限 tol。默认时，tol 为

```
max(size(A))*norm(A, 1)*eps
```

● [x, w] = nnls(A, b) 还返回二重向量 w，其中当 $x_i = 0$ 时，$w_i > 0$；当 $x_i > 0$ 时，$w_i < 0$。

● [x, w] = nnls(A, b, tol) 求解方程组，返回二重向量 w，指定容限 tol。

【例 5-7】 对于一个 4×2 的问题，比较无约束最小二乘解与 nnls 函数的解：

```
A =
    0.0372   0.2869
    0.6861   0.7071
    0.6233   0.6245
    0.6344   0.6170
b =
    0.8587
    0.1781
    0.0747
    0.8405
```

```
[A\b nnls(A,b)] =
    -2.5627        0
     3.1108   0.6929
[norm(A*(a\b)-b) norm(A*nnls(a,b)-b)] =
     0.6674 0.9118
```

nnls 函数的解拟合程度不是很好，但没有负的成分。

5.2　决策树

在做非线性最小二乘分析时，我们假设已知自变量和应变量之间的关系式。现在假设不知道它们之间的关系式，而且不愿意用线性模型去拟合它们之间的关系，就需要采用一种非参数类型的回归拟合方法。基于“树”的方法是其中之一。一个决策树由一系列可以用“是”或“不是”来回答的问题加上一系列拟合响应值组成。每个问题询问自变量是否满足给定的条件。自变量可以是连续的，也可以是离散的，这取决于问题的答案。根据回答的结果，要么进入下一个问题（不满足条件时），要么得到一个拟合响应值（满足条件时）。

MATLAB 统计工具箱提供了进行决策分析的函数。下面结合 MATLAB 自带的例子进行介绍。

在本例中，用决策树拟合 carsmall 数据集中的变量。这里采用一个连续的自变量（car weight）和一个离散的自变量（model year），目的是将 MPG 作为 car weight 和 model year 的函数。

首先装载数据，然后创建自变量值的矩阵（x）和响应变量向量（y）。把 model year 列指定为分类变量，拟合决策树。在本数据集中，有来源于 3 个不同型号、年份的小汽车，分别是 1970，1976 和 1982 年。

```
load carsmall
x = [Weight,Model_Year];
y = MPG;
t = treefit(x,y,'catidx',2);
treedisp(t,'name',{'Wt' 'Yr'});
```

现在有一辆重 3000 磅的 1982 年型小汽车，试用本模型确定预测变量 mileage（哩数）的值。如图 5-3 所示，从顶层节点开始，重量小于阈值 3085.5，所以选择左侧的分支。型号不是 1970 年，也不是 1976 年，所以选择右边的分支。继续向下移动，直到到达给出预测值的终节点。本例中，预测值为 38 里每加仑。可以用 treeval 函数找到任何系列的自变量值的拟合值。

```
treeval(t, [3000 82])
ans =
    38
```

使用类似于这种具有很多分支的树有一个缺点，即它可能对当前数据拟合得很好，对新数据却不一定能准确拟合。它的某些下级分支可能较强地受到异常值和其他因素的影响。如果可能的话，我们宁愿找到一种简单的树结构，避免“over-fitting（过拟合）”问题。下面通过交叉验证（cross-validation）来估计最好的树的大小。首先，为一系列更简单的树和本树给出一个剩余方差的“resubstitution（重替换）”估计，并在图中用蓝线绘图。本估计可能低估了真实的剩余方差。然后，做相同对象的“cross-validation”估计并在蓝线上方用红线绘图。

交叉验证过程还提供了一个剪枝水平的估计，以获得树的最佳大小。

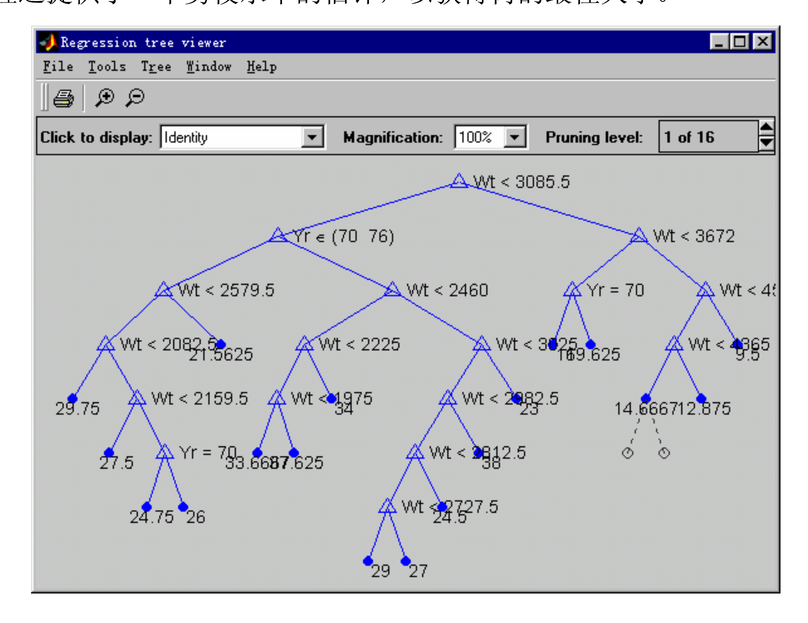


图 5-3　根据示例数据得到的决策树

```
 [c,s,ntn] = treetest(t,'resub');
[c2,s2,n2,best] = treetest(t,'cross',x,y);
plot(ntn,c,'b-', n2,c2,'r-', n2(best+1),c2(best+1),'mo');
xlabel('Number of terminal nodes')
ylabel('Residual variance')
legend('Resubstitution error','Cross-validation error','Estimated best
        tree size')
best
best =
    10
```

生成的图形如图 5-4 所示。

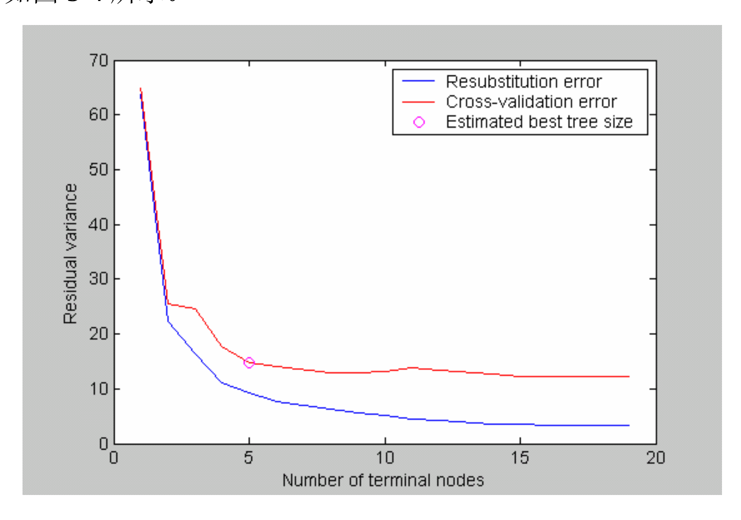


图 5-4　估计树的最佳大小

最好的树是那个剩余方差沿交叉验证线与最小值相比，向上不超过一个标准差的树。本例中，剩余方差正好超过了 14。输出 best 的值从 0 开始（表示没有剪枝），所以给它加 1，作为编号输入到其他输出变量中。

```
c2(best+1)
ans =
    14.3440
```

使用输出 best 创建一个更小的树，它达到了我们估计的最佳大小。

```
t0 = treeprune(t,'level',best);
treedisp(t0,'name',{'Wt' 'Yr'})
```

生成的更小的树如图 5-5 所示。

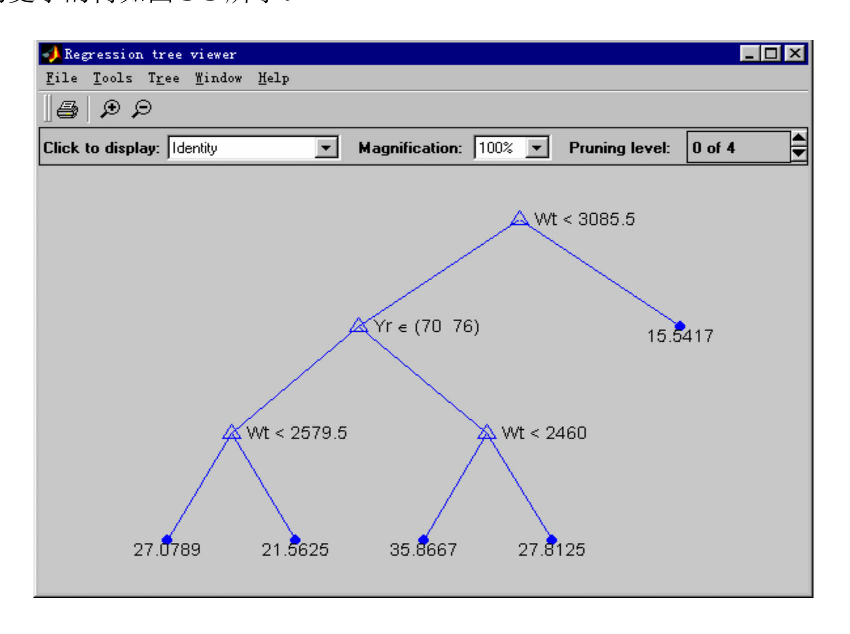


图 5-5　更小的树

现在绘原始数据的图，并覆盖通过本树获得的拟合值。注意，本树不区分 1970 年或 1976 年的车，所以，创建一个包含 1976 年车的拟合值的向量（yold）和 1982 年车拟合值的向量（ynew）。1970 年车与 1976 年车具有相同的拟合值。

```
xx = (1500:20:5000)';
ynew = treeval(t0,[xx 82*ones(size(xx))]);
yold = treeval(t0,[xx 76*ones(size(xx))]);
gscatter(Weight,MPG,Model_Year,'rgb','osx');
hold on; plot(xx,yold,'b:', xx,ynew,'r--'); hold off
```

生成的散点图如图 5-6 所示。

tree 函数族（treedisp，treefit，treeprune，treetest 和 treeval）还可以接受一个分类响应变量。此时，树的拟合值是具有最高预测概率的类。该概率是预测值落在给定节点上的概率。

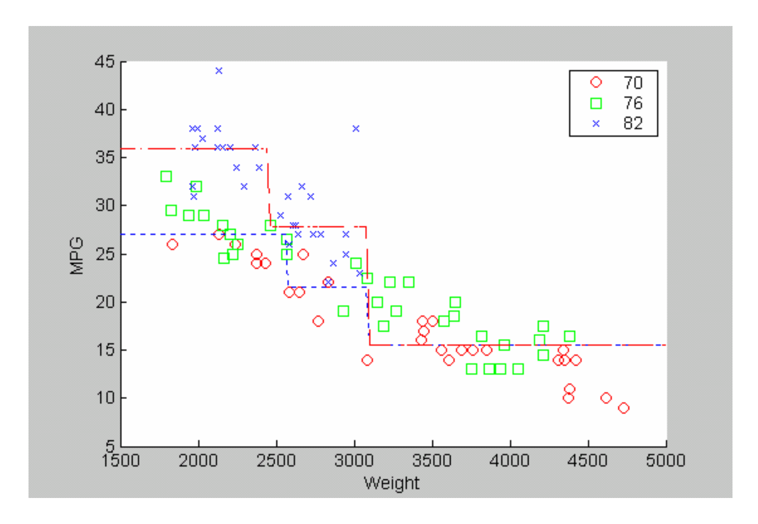


图 5-6　Weight-MPG 散点图

第 6 章　假 设 检 验

运用一定的统计方法进行数据分析时，常常要求数据满足一定的条件，如正态性、方差齐性、独立性等。数据是否满足假设，需要检验。

在总体分布函数完全未知或只知分布形式，但不知其参数时，为了推断总体的某些性质，需要提出关于总体的假设。假设是否合理，需要检验。

假设检验中有几个概念需要掌握。

1．原假设（零假设）、备择假设

假设现在在显著性水平α下，检验总体均值 μ 是否等于样本均值 μ_0，即检验假设

$$H_0: \mu = \mu_0 \qquad\qquad H_1: \mu \neq \mu_0$$

则称 H_0 为原假设（或零假设），称 H_1 为备择假设。

2．拒绝域、临界点

当检验统计量取某个区域中的值时，拒绝原假设，则称该取值区域为拒绝域，称拒绝域的边界点为临界点。

3．第 1 类错误、第 2 类错误

当零假设实际上为真，却拒绝零假设时所犯的错误称为“弃真”错误，或第 1 类错误；当零假设实际上不为真，却接受零假设时所犯的错误称为“取伪”错误，或第 2 类错误。

4．双边检验、单边检验

对于上面的假设问题，如果备择假设表示μ可能大于μ_0，也可能小于μ_0，则称这种假设检验为双边检验；如果备择假设表示μ可能大于μ_0，或可能小于μ_0，则称这种假设检验为单边检验。

6.1　单个样本的 t 检验

6.1.1　基本数学原理

t 检验是用小样本检验总体参数，特点是在均方差不知道的情况下，可以检验样本平均数的显著性。

对于单个正态总体并且方差未知的情况，用下面的统计量来检验其平均数的显著性（假设样本均值与总体均值相等，即$\mu=\mu_0$）。

$$T = \sqrt{n}\frac{\overline{X} - \mu_0}{S}$$

式中，S 为样本的方差，$\overline{X}$ 为样本均值，μ_0 为总体均值，n 为样本大小。

当原假设成立时，上面的统计量应该服从自由度为 $n-1$ 的 t 分布。

6.1.2　有关函数介绍

用 ttest 函数进行样本均值的 t 检验，其调用格式为：

● ttest(x,m) 在 0.05 的显著性水平上进行 t 检验，以确定在标准差未知的情况下取自正态分布的样本的均值是否为 m。

● h = ttest(x,m,alpha) 给出显著性水平的控制参数 alpha。例如，当 alpha = 0.01 时,如果 h=1，则在 0.01 的显著性水平上拒绝零假设；若 *h*=0，则不能在该水平上拒绝零假设。

● [h,sig,ci] = ttest(x,m,alpha,tail) 允许指定是进行单侧检验还是进行双侧检验。tail 可以有下面 3 个取值：

tail = 0 (为默认设置)指定备择假设 $\bar{x} \neq \mu$ 。

tail = 1 指定备择假设 $\bar{x} > \mu$ 。

tail = -1 指定备择假设 $\bar{x} < \mu$ 。

sig 为与 t 统计量有关的 *p* 值。sig 为零假设为 $x=\mu$ 时 T 的观测值大于或等于 μ 的概率。

ci 为均值真值的 1-alpha 置信区间。

6.1.3 应用实例

【例 6-1】 测得一批钢件的 20 个样品的屈服点（单位：T/mm^2）为：

```
4.98  5.11  5.20  5.11  5.00  5.61  4.88  5.27  5.38  5.20
5.46  5.27  5.23  4.96  5.35  5.15  5.35  4.77  5.33  5.54
```

并且假设屈服点服从正态分布。已知总体均值为 5.20，试对该样本的数据进行均值检验。假设该样本的均值与总体均值之间没有显著差别。

```
x=[4.98 5.11 5.20 5.11 5.00 5.61 4.88 5.27 5.38 5.20 5.46 5.27 5.23 4.96…
   5.35 5.15 5.35 4.77 5.33 5.54];
[h,sig,ci] = ttest(x,0)
h =
    1
sig =
    0
ci =
    5.1052   5.3098
```

h = 1，拒绝零假设，认为该样本的均值与总体均值之间有显著差异。显著性水平为 0，表示在 100 次试验中，如果样本均值等于总体均值，则 T 值大于或等于总体均值的次数为 0 次，它小于 0.05，同样支持前面的结论。均值的 95%置信区间为[5.1052 5.3098]。

6.2 两个样本的 t 检验

6.2.1 基本数学原理

进行两个独立正态总体下样本均值的比较时，根据方差齐与不齐两种情况，应用不同的统计量进行检验。

方差不齐时，检验统计量为

$$T = \frac{\overline{X} - \overline{Y}}{\sqrt{\frac{S_X^2}{m} + \frac{S_Y^2}{n}}}$$

式中，$\overline{X}$ 和 $\overline{Y}$ 表示样本 1 和样本 2 的均值；S_X^2 和 S_Y^2 为样本 1 和样本 2 的方差；m 和 n 为样本

1 和样本 2 的数据个数。

方差齐时，检验统计量为

$$T=\frac{\overline{X}-\overline{Y}}{S_w\sqrt{\frac{1}{m}+\frac{1}{n}}}$$

式中，S_W 为两个样本的标准差，它是样本 1 和样本 2 的方差的加权平均值的平方根：

$$S_w=\sqrt{\frac{(m-1)S_X^2+(n-1)S_Y^2}{m+n+1}}$$

当两个总体的均值差异不显著时，该统计量应服从自由度为 $m+n-2$ 的 t 分布。

6.2.2　有关函数介绍

ttest2 函数对两个样本的均值差异进行 t 检验，其调用格式为：

- h = ttest2(x, y) 假设 x 和 y 为取自服从正态分布的两个样本。在它们的标准差未知但相等时检验它们的均值是否相等。当 h=1 时，可以在 0.05 的水平上拒绝零假设；当 h=0 时，在该水平上接受零假设。
- [h, significance, ci] = ttest2(x, y, alpha) 给出显著性水平的控制参数 alpha。例如，当 alpha = 0.01 时,如果 h=1，则在 0.01 的显著性水平上拒绝零假设；若 h=0，则不能在该水平上拒绝零假设。此处，ci 为均值差异真值的 100(1-alpha)%置信区间。significance 参数是与 t 统计量相关的 p 值。它是零假设为 x 与 y 的均值相等时 T 的测量值相等或更大的概率。ci 为均值差异真值的 1-alpha 置信区间。
- ttest2(x, y, alpha, tail) 允许指定进行单侧检验或双侧检验。tail 可以有下面 3 个取值：

  tail = 0 (默认设置)　指定备择假设 $\bar{x}\neq\mu$；

  tail = 1　指定备择假设 $\bar{x}>\mu$；

  tail = -1　指定备择假设 $\bar{x}<\mu$。

6.2.3　应用实例

【例 6-2】　对两种不同的水稻品种 A，B 分别统计了 8 个地区的单位面积产量（单位：kg）。

品种 A	86	87	56	93	84	93	75	79
品种 B	80	79	58	91	77	82	76	66

要求检验两个水稻品种的单位面积产量之间是否有显著差异。

```
x=[86 87 56 93 84 93 75 79];
y=[80 79 58 91 77 82 76 66];
[h,significance,ci] = ttest2(x,y)
h =
     0
significance =
     0.3393
ci =
    -6.4236   17.4236
```

h = 0, 表示不能拒绝零假设，认为两种水稻品种的单位面积产量之间没有显著差异。显著性水平 significance=0.3393，表示在 100 次试验中，如果两种水稻品种的单位面积产量相等，

则约有 34 次 t 统计量大于或等于均值差。均值差的 95%置信区间为[-6.4236　17.4236]。

6.3　z 检验

z 检验在方差已知的情况下检验数据是否服从给定均值的正态分布。用 ztest 函数在给定方差的条件下进行均值检验，其调用格式为：

- ztest(x, m, sigma) 在 0.05 的水平上进行 z 检验，以确定服从正态分布的样本的均值是否为 m，标准差是否为 sigma。
- h = ztest(x, m, sigma, alpha) 给出显著性水平控制参数 alpha。例如，若 alpha = 0.01, 则当结果 h=1 时,可以在 0.01 的显著性水平上拒绝零假设。若 $h = 0$，则不能在该水平上拒绝零假设。
- [h, sig, ci] = ztest(x, m, sigma, alpha, tail) 允许指定进行单侧检验或双侧检验。tail 参数可以有下面几个取值：

tail = 0 (默认设置)　指定备择假设 $\bar{x} \neq \mu$ 。

tail = 1　指定备择假设 $\bar{x} > \mu$ 。

tail = -1　指定备择假设 $\bar{x} < \mu$ 。

sig 为与 z 统计量相关的 p 值，$z = \dfrac{\bar{x}-\mu}{\sigma}$ 。sig 是零假设为 $x=\mu$ 时，z 的观测值大于或等于 μ 的概率。ci 为均值真值的 1-alpha 置信区间。

【例 6-3】　某批矿砂的 5 个样品中的镍含量，经测定为（%）

3.25　3.27　3.24　3.26　3.24

设测定值总体服从正态分布，方差为 0.04，问在 0.01 的水平上能否接受假设。这批矿砂的镍含量的均值为 3.25。

```
x=[3.25 3.27 3.24 3.26 3.24];
[h,sig,ci] = ztest(x,3.25,0.04,0.01)
h =
    0
sig =
    0.9110
ci =
    3.2059   3.298
```

h=0, sig>0.01，所以接受零假设，认为在 0.01 的水平上可以认为这批矿砂的镍含量的均值为 3.25。

【例 6-4】　下面列出的是某工厂随机选取的 20 只部件的装配时间（分）：

9.8　10.4　10.6　9.6　9.7　9.9　10.9　11.1　9.6　10.2

10.3　9.6　9.9　11.2　10.6　9.8　10.5　10.1　10.5　9.7

设装配时间的总体服从正态分布，方差为 0.4，是否可以认为装配时间的均值在 0.05 的水平上显著地大于 10。

```
x=[9.8  10.4  10.6  9.6  9.7  9.9  10.9  11.1  9.6  10.2…
10.3  9.6  9.9  11.2  10.6  9.8  10.5  10.1  10.5  9.7];
```

%由于是单侧检验，需要指定 tail 值。

```
[h,sig,ci] = ztest(x,10,0.4,0.05,1)
```

```
h =
    1
sig =
    0.0127
ci =
   10.0529      Inf
```

检验结果显示，h=1, sig < 0.05，所以拒绝零假设，认为不可以认为装配时间的均值在 0.05 的水平上显著大于 10。

第 7 章　分布的检验

进行数据分析时，许多情况下首先要假设数据服从一定形式的分布。例如工程地质学中，为了进一步研究岩体的结构，进行均质区域的划分，并提供地下水渗流和岩体力学参数，研究人员进行野外裂隙测量以后，利用取得的样本进行总体的裂隙网络模拟。该方法就需要研究裂隙样本中裂隙走向、长度、宽度、间距等参数服从什么样的分布，然后利用对应分布来生成网络，提供进一步研究的基础。常用的做法是假设裂隙参数服从某种分布，然后进行检验。

常用的分布检验方法有非参数法（如卡方优度检验、Kolmogorov-Smirnov 检验、符号检验等）和图示法（包括 q-q 图、p-p 图、直方图）等。

7.1　Jarque-Bera 检验

7.1.1　基本数学原理

Jarque-Bera 检验评价 ***X*** 服从未知均值和方差的正态分布的假设是否成立。该检验基于 ***X*** 的样本偏度和峰度。对于正态分布数据，样本偏度接近于 0，样本峰度接近于 3。Jarque-Bera 检验确定样本偏度和峰度是否与它们的期望值相差较远。

Jarque-Bera 检验不能用于小样本的检验。对于小样本，用 Lilliefors 检验比较合适。

7.1.2　有关函数介绍

可用 jbtest 函数进行 Jarque-Bera 检验，测试数据对正态分布的拟合程度，其调用格式为：

- H = jbtest(X) 对输入数据向量 ***X*** 进行 Jarque-Bera 检验，返回检验结果 *H*。如果拒绝 ***X*** 服从正态分布的假设，则 *H*=1；否则 *H*=0。如果检验在 5%的水平上显著，则拒绝假设。
- H = jbtest(X, alpha) 在 100*alpha%的水平上进行 Jarque-Bera 检验，而不是在 5%的水平上。其中 alpha 必须界于 0 和 1 之间。
- [H, P, JBSTAT, CV] = jbtest(X, alpha) 返回 3 个其他输出。P 为检验的 *p* 值，JBSTAT 为检验的统计量，CV 为确定是否拒绝零假设的临界值。

【例 7-1】　检验汽车重量数据是否服从正态分布。

```
load carsmall
[h,p,j] = jbtest(Weight)
h =
    1
p =
    0.026718
j =
    7.2448
```

p 值为 2.67%，所以拒绝数据服从正态分布的零假设。通过对数转换，分布更接近正态分布，但仍然在 5%的水平上具有显著差异。

```
[h,p,j] = jbtest(log(Weight))
h =
    1
p =
    0.043474
j =
    6.2712
```

7.1.3 应用实例

【例 7-2】 下面列出了 84 个伊特拉斯坎（Etruscan）人男子的头颅的最大宽度（mm），试检验这些数据是否处于正态分布。

```
141  148  132  138  154  142  150  146  155  158
150  140  147  148  144  150  149  145  149  158
143  141  144  144  126  140  144  142  141  140
145  135  147  146  141  136  140  146  142  137
148  154  137  139  143  140  131  143  141  149
148  135  148  152  143  144  141  143  147  146
150  132  142  142  143  153  149  146  149  138
142  149  142  137  134  144  146  147  140  142
140  137  152  145
```

在命令窗口输入下面的命令行进行检验：

```
X=[141  148  132  138  154  142  150  146  155  158…
   150  140  147  148  144  150  149  145  149  158…
   143  141  144  144  126  140  144  142  141  140…
   145  135  147  146  141  136  140  146  142  137…
   148  154  137  139  143  140  131  143  141  149…
   148  135  148  152  143  144  141  143  147  146…
   150  132  142  142  143  153  149  146  149  138…
   142  149  142  137  134  144  146  147  140  142…
   140  137  152  145];
[H,P,JBSTAT,CV] = jbtest(X)
H =
   0
P =
   0.7610
JBSTAT =
   0.5461
CV =
   5.9915
```

H=0, p>0.5，所以接受零假设，认为上面的数据服从正态分布。检验统计量为 0.5461，拒绝零假设的临界值为 5.9915，检验统计量的值比临界值小。

7.2 单样本的 Kolmogorov-Smirnov 检验

7.2.1 基本数学原理

该检验为拟合优度型检验，可以检验样本数据是否服从指定的理论分布。假设 $F_0(x)$是一

已知的分布函数，$F_n(x)$是对未知的总体分布函数 $F(x)$的一个较优的估计。取检验统计量

$$D=\max|F_n(x)-F_0(x)|$$

则样本数据服从指定分布（即 $F(x)=F_0(x)$）时，D 的观测值应该较小。如果 D 的观测值较大，则零假设不成立。

7.2.2 有关函数介绍

用 kstest 函数进行单样本分布的 Kolmogorov-Smirnov 检验，其调用格式为：

● H = kstest(X) 进行 Kolmogorov-Smirnov 检验，确定数据向量 X 中的值是否服从标准正态分布。零假设为 X 服从标准正态分布。如果拒绝零假设，则 H=1；否则 H=0。如果检验在 5%的水平上显著，则拒绝零假设。

kstest 函数将所有 x 值中的最大差异作为它的检验统计量。在数学上，它可以写成：

$$\max(|F(x)-G(x)|)$$

式中，$F(x)$ 为 X 的值小于或等于 x 的比例，$G(x)$ 为 x 处的标准正态累加分布函数。

● H = kstest(X, cdf) 比较 X 的分布与由两列矩阵 cdf 定义的假设分布。第 1 列包含一系列可能的 x 值，第 2 列包含对应的假设累加分布函数值。如果可能，应该定义累加分布函数，使得第 1 列包含 X 中的值。如果 X 中的值没有在累加分布函数的第 1 列被发现，则 kstest 函数将通过内插进行近似。X 中的所有值必须位于累加分布函数第 1 列的最大值和最小值之间。如果第 2 个变量为空（cdf=[]），则 kstest 函数使用标准正态分布，就像没有第 2 个变量一样。

Kolmogorov-Smirnov 检验需要预先给定累加分布函数。如果累加分布函数是从数据估计得到的，则该检验不够精确。在没有指定参数的情况下检验数据是否服从正态分布，用 Lilliefors 检验效果更佳。

● H = kstest(X, cdf, alpha, tail) 指定显著性水平 alpha 和备择假设的代码 tail。如果 tail = 0（默认选项），则 kstest 进行双边检验 $F \neq G$；如果 tail = -1，则备择假设为 $F < G$，如果 tail=1，则备择假设为 $F > G$。检验统计量的形式取决于 tail 的值：

tail = 0: $\max(|F(x)-G(x)|)$

tail = -1: $\max(|G(x)-F(x)|)$

tail = 1: $\max(|F(x)-G(x)|)$

● [H, P, KSSTAT, CV] = kstest(X, cdf, alpha, tail) 返回观测的 p 值 P、观测的 Kolmogorov-Smirnov 统计量 KSSTAT 和决定 KSSTAT 是否显著的临界值 CV。如果 CV 的返回值为 NaN，则 kstest 函数根据假设公式计算 p 值，而不是通过比较 KSSTAT 与临界值来确定显著性。

7.2.3 应用实例

【例 7-3】 现在生成一些等间隔的数字，并进行 Kolmogorov-Smirnov 检验，看它们拟合正态分布的程度。

```
x = -2:1:4
x =
   -2   -1    0    1    2    3    4
[h,p,k,c] = kstest(x,[],0.05,0)
h =
```

```
          0
p =
          0.13632
k =
          0.41277
c =
          0.48342
```

尽管这些数据看起来不服从正态分布，但由于 *p*=0.13632，远大于 0.05，所以接受数值服从标准正态分布的零假设。

为了理解该检验，现在生成一个经验累加分布函数，并将理论正态分布曲线叠加在上面。

```
xx = -3:.1:5;
cdfplot(x)
hold on
plot(xx,normcdf(xx),'r--')
```

生成的图形如图 7-1 所示。

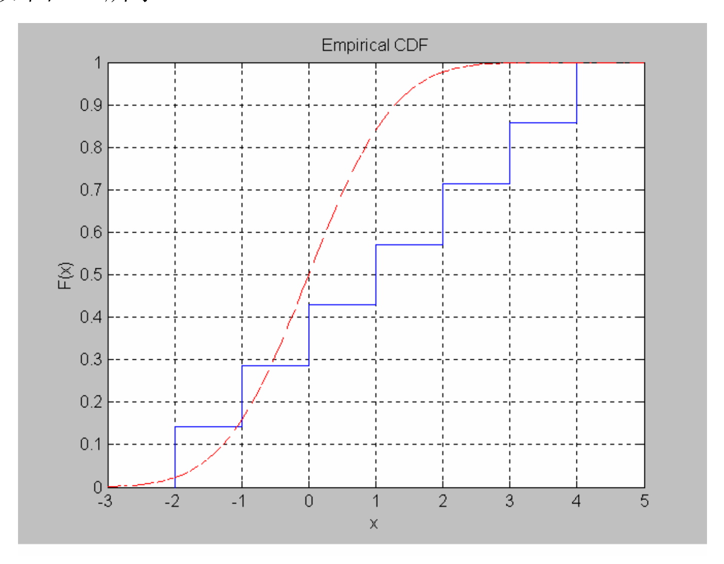


图 7-1 经验累加分布函数图叠加理论正态分布曲线

Kolmogorov-Smirnov 检验统计量为这些曲线之间的最大差异。图 7-1 中最大差异为 0.41277，发生在接近 $x = 1.0$ 的地方。该处经验曲线的值为 3/7，可以很方便地算出此处的曲线差值。

```
normcdf(1) - 3/7
ans =
          0.41277
```

还可以进行单侧检验。下面设置 tail = -1 然后进行计算。

```
[h,p,k] = kstest(x, [], .05, -1)
h =
        0
p =
        0.068181
k =
```

```
     0.41277
```

检验统计量与前面的相同，因为都是在 $x = 1.0$ 处，但单侧检验的 p 值比前面的小。如果进行另一侧的检验，令 x=-1.0，会发现检验统计量改变了，差异最大值位于 $x = -1.0$ 附近。

```
[h,p,k] = kstest(x,[],0.05,1)
h =
     0
p =
     0.77533
k =
     0.12706
2/7 - normcdf(-1)
ans =
     0.12706
```

【例 7-4】 现在生成服从威布尔分布的随机数，并检验威布尔分布和指数分布。

```
x = weibrnd(1,2,100,1);
kstest(x,[x weibcdf(x,1,2)])
ans =
    0
kstest(x,[x expcdf(x,1)])
ans =
    1
```

可见，当检验威布尔分布时，ans=0；当检验指数分布时，ans=1。所以原数据服从威布尔分布，而不服从指数分布。

【例 7-5】 在一大批相同型号的电子元件中随机地抽取 10 只做寿命试验，测得它们的使用寿命（单位：小时）为

```
420    500    920    1380    1510
1650   1760   2100   2320    2350
```

试问电子元件的寿命是否服从均值为 1500 小时的指数分布？

```
x=[420  500  920  1380  1510  1650  1760  2100  2320  2350]';
kstest(x, [x expcdf(x, 1500)])
ans =
    0
```

所以，接受原假设，认为电子元件的寿命服从均值为 1500 小时的指数分布。

7.3 两个样本的 Kolmogorov-Smirnov 检验

7.3.1 基本数学原理

该检验可用来检验两个独立样本是否取自同一总体。进行检验时，对每个观察样本做累积频数分布，并对分布采用相同的间隔。对于每个间隔。将两个阶梯函数相减，并着重分析观测值的差值中间的最大者。

对于单尾检验，令

$$D=\max[F_{n1}(x)-F_{n2}(x)],$$

对于双尾检验，令

$$D=\max|F_{n1}(x)-F_{n2}(x)|$$

其中，$F_{n1}(x)$和 $F_{n2}(x)$分别为两个样本的观测累积阶梯函数。如果 D 较小，则可以认为两个样本取自同一个总体；如果 D 较大，大到一定程度，则可以认为两个样本并不是取自同一总体。

7.3.2 有关函数介绍

用 kstest2 函数进行 Kolmogorov-Smirnov 检验，比较两个样本之间的分布，其调用格式为：

● H = kstest2(X1, X2) 进行双样本 Kolmogorov-Smirnov 检验，比较两个数据向量 X1 和 X2 中值的分布。该检验的零假设为 X1 和 X2 具有相同的连续分布。备择假设为它们具有不同的连续分布。如果分布相同，则 H=1；否则 H=0。如果检验在 5%的水平上显著，则拒绝假设。

对于每个潜在的 x 值，Kolmogorov-Smirnov 检验对 X1 的值小于 x 的比例和 X2 的值小于 x 的比例之间进行比较。kstest2 函数使用所有 x 值中的最大差异作为它的检验统计量。在数学上，它可以写作

$$\max(|F1(x)-F2(x)|)$$

式中，$F1$ 为 X1 的值小于或等于 x 的比例，$F2$ 为 X2 的值小于或等于 x 的比例。

● H = kstest2(X1,X2,alpha,tail) 指定显著性水平 alpha 和备择假设的代码 tail。如果 tail = 0（默认选项），则 kstest 进行双边检验 $F1\neq F2$；如果 tail = -1，则备择假设为 $F1<F2$；如果 tail=1，则备择假设为 $F1>F2$。检验统计量的形式取决于 tail 的值：

tail = 0: $\max(|F1(x)-F2(x)|)$

tail = -1: $\max(|F2(x)-F1(x)|)$

tail = 1: $\max(|F1(x)-F2(x)|)$

● [H, P, KSSTAT, CV] = kstest2(X, cdf, alpha, tail) 还返回观测的 p 值 P、观测的 Kolmogorov-Smirnov 统计量 KSSTAT 和决定 KSSTAT 是否显著的临界值 CV。如果 CV 的返回值为 NaN，则 kstest 函数根据假设公式计算 p 值，而不是通过比较 KSSTAT 与临界值来确定显著性。

7.3.3 应用实例

【例 7-6】 现在比较等间隔小样本与服从正态分布的大样本。

```
x = -1:1:5
y = randn(20,1);
[h,p,k] = kstest2(x,y)
h =
    1
p =
    0.0403
k =
    0.5714
```

从计算结果可以看出，二者的分布差异在 5%的水平上是显著的。为了使显示差异可视化，下面将两个经验累加分布函数叠加到图中。Kolmogorov-Smirnov 统计量为这些函数之间的最大差异。改变某条曲线的颜色和线型以后，将发现最大差异位于 x = 1.9 附近。

```
cdfplot(x)
hold on
cdfplot(y)
h = findobj(gca,'type','line');
set(h(1),'linestyle',':','color','r')
1 - 3/7
ans =
0.5714
```

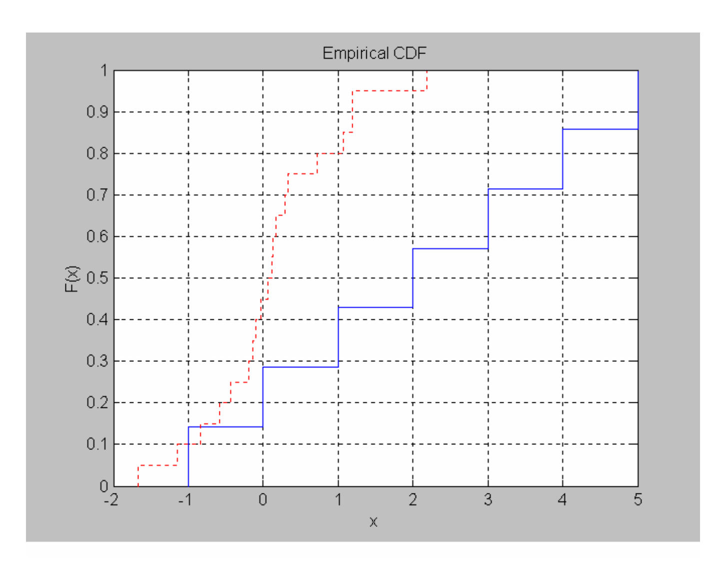


图 7-2　叠加两个经验累加分布函数图

【例 7-7】　Lepley 曾把 10 名 7 年级学生和 10 名 11 年级学生的系统学习作过比较。他假设，在学习同一个课程时，较年轻的 7 年级学生记住先前学的材料比 11 年级学生要差一些。为检验此假设，他将两个组在该课程中学过的材料的前一半所犯错误的百分比进行比较，得到如下数据：

7　年级学生	39.1	41.2	45.2	46.2	48.4	48.7	55.0	40.6	52.1	47.2
11 年级学生	35.2	39.2	40.9	38.1	34.4	29.1	41.8	24.3	32.4	32.6

假设 7 年级学生犯错误的百分比比 11 年级学生的要高一些。

```
x=[39.1 41.2 45.2 46.2 48.4 48.7 55.0 40.6 52.1 47.2]';
y=[35.2 39.2 40.9 38.1 34.4 29.1 41.8 24.3 32.4 32.6]';
```

%由于采用单侧检验，需要指定 tail 值。

```
[h,p,k] = kstest2(x,y,0.05 1)
h=
    0
p =
    1
k =
    0
```

h=0，p>0.05，所以接受零假设，认为 7 年级学生犯错误的百分比比 11 年级学生的要高一些，即较年轻的 7 年级学生记住先前学的材料的能力比 11 年级学生要差一些。

7.4 Lilliefors 检验

7.4.1 基本数学原理

Lilliefors 检验评价样本 *X* 服从均值和方差未知的正态分布的零假设是否成立，对应的备择假设为 *X* 不服从正态分布。本检验比较 *X* 的经验分布与具有相同均值和方差的正态分布。它近似于 Kolmogorov-Smirnov 检验，但它进行检验时，正态分布的参数是从对 *X* 的估计得到的，而不是事先指定的。

7.4.2 有关函数介绍

用 lillietest 函数进行正态分布的 Lilliefors 检验，其调用格式为：

● H = lillietest(X) 对输入数据向量 ***X*** 进行 Lilliefors 检验，返回假设检验的结果 H。如果拒绝 ***X*** 服从正态分布的零假设，则 *H*=1；否则 *H*=0。如果检验在 5%的水平上显著，则拒绝假设。

● H = lillietest(X, alpha) 在 100*alpha%的水平上，而不是在 5%的水平上进行 Lilliefors 检验。alpha 必须在 0.1 和 0.2 之间。

● [H, P, LSTAT, CV] = lillietest(X, alpha) 返回 3 个其他输出。P 为检验的 *p* 值，通过在一系列由 Lilliefors 创建的表中进行插值得到。LSTAT 为检验统计量的值。CV 为确定是否拒绝零假设的临界值。如果 LSTAT 的值位于 Lilliefors 表之外，则 P 中返回 NaN（空值），但 H 显示是否拒绝假设。

7.4.3 应用举例

【例 7-8】 下面利用 carsmall 数据检验汽车的重量是否服从正态分布。

```
load carsmall
[h p l c] = lillietest(Weight);
[h p l c]
ans =
   1.0000    0.0232    0.1032    0.0886
```

Lilliefors 检验的统计量为 0.10317，比 5%水平检验的临界值 0.0886 大，所以拒绝正态分布的假设。实际上，本检验的 *p* 值接近于 0.02。

下面用直方图显示分布。从图 7-3 中可以看出，直方图从 2250 顶点处向右偏斜，频率向左侧急剧下降，但在右侧则是缓慢下降的。

```
hist(Weight)
```

有时对变量进行转换可使分布更加接近正态。利用对数转换，可使向右侧的偏斜更小。

```
[h p l c] = lillietest(log(Weight))
ans =
          0     0.13481    0.077924      0.0886
```

现在 *p* 值接近于 0.13，所以不必拒绝零假设。

利用转换以后的数据绘直方图：

```
hist(log(Weight))
```

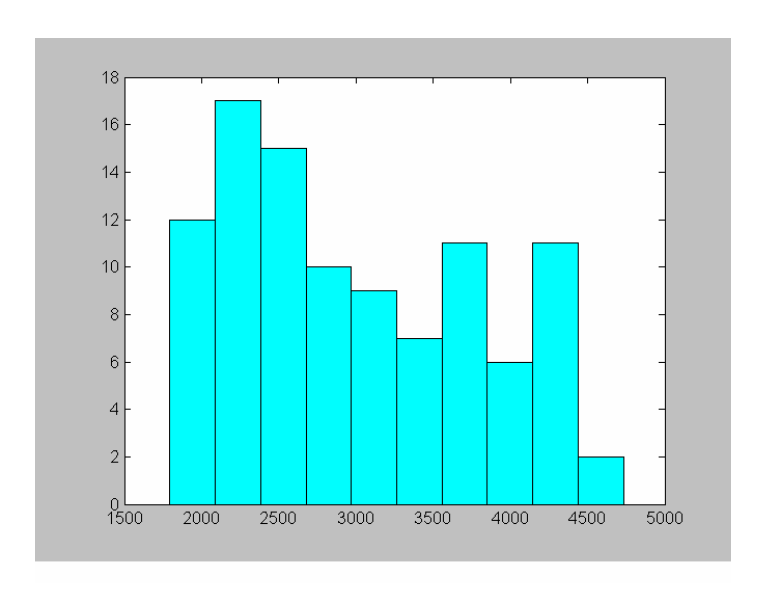


图 7-3　直方图

生成的图形如图 7-4 所示。

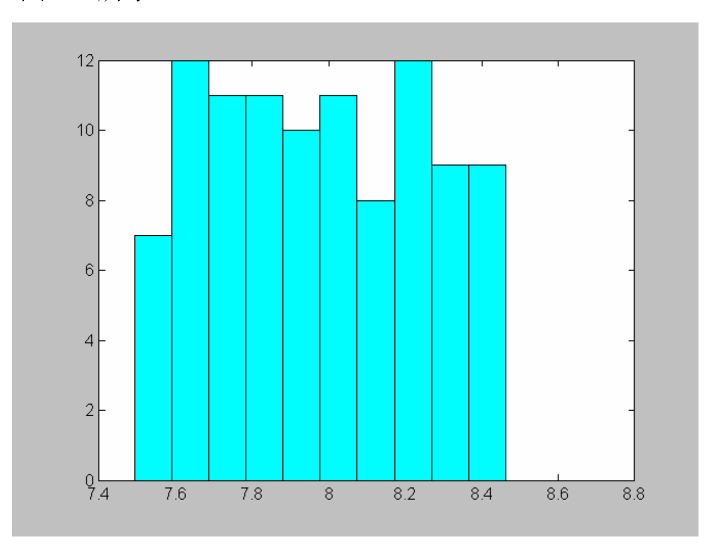


图 7-4　转换数据以后生成的直方图

【例 7-9】　如果一个矩形的宽度和长度的比接近 0.618，则称为黄金矩形。这种尺寸的矩形使人们看上去有良好的感觉，现代的建筑构件、工艺品等都采用黄金矩形。下面列出某工艺品工厂随机取的 20 个矩形的宽度与长度的比值，检验该工厂生产的矩形的宽度与长度的比值是否服从正态分布。

```
0.693  0.749  0.654  0.670  0.662  0.672  0.615  0.606  0.690  0.628
0.668  0.611  0.606  0.609  0.601  0.553  0.570  0.844  0.576  0.933
```

在命令窗口中输入下面的命令行：

```
x=[0.693  0.749  0.654  0.670  0.662  0.672  0.615  0.606  0.690  0.628...
   0.668  0.611  0.606  0.609  0.601  0.553  0.570  0.844  0.576  0.933]';
[h p l c] = lillietest(x);
```

```
h =
    1
p =
    0.0279
l =
    0.2127
c =
    0.1900
```

h=1，p<0.05，所以拒绝零假设，认为上面的数据不服从正态分布。检验统计量 l=0.2127，拒绝零假设的临界值等于 0.190，所以检验统计量的值比临界值大，同样拒绝零假设。

第 8 章　非参数检验

非参数方法可以广泛应用于社会科学、行为科学、生物科学和数理科学等研究领域。与参数方法相比，它具有分布自由、可用于按数值意义讲并不严格但有一定等级顺序的资料的分析以及计算简单三大优点。

8.1　Kruskal-Wallis 检验

8.1.1　基本数学原理

该检验用来检验 k 个独立样本是否来自不同总体，若这 k 个样本服从相同分布，则在样本容量不太小的情况下，下面的统计量 H 服从自由度 k-1 的 χ^2 分布：

$$H = \frac{12}{N(N+1)}\sum_{j=1}^{k}\frac{R_i^2}{n_j} - 3(N+1)$$

式中，k 为样本数，n_j 为第 j 个样本中的案例数，N 为所有样本的案例数之和，R_j 为第 j 个样本（列）中的秩和。

该法是 Mann-Whitney U 检验的推广，它不要求数据服从正态分布，因而在一定情况下可以代替 F 检验。

8.1.2　有关函数介绍

可用 kruskalwallis 函数进行单因素方差分析的 Kruskal-Wallis 非参数检验，其调用格式为：

● p = kruskalwallis(X) 进行 Kruskal-Wallis 检验，比较 $m\times n$ 矩阵 $\boldsymbol{X}$ 的列均值，其中，每一列代表一个独立样本，包含 m 个相互独立的观测值。Kruskal-Wallis 检验是典型单因素方差分析的非参数版本。该函数返回 $\boldsymbol{X}$ 中的所有样本取自相同总体（或取自具有相同均值的不同总体）的零假设的概率。

如果 p 值接近于 0，则可以怀疑零假设，认为至少有一个样本均值与其他样本均值之间有显著差异。临界值 p 值的选择决定结果是否“统计上显著”，该值的大小由用户朋友给定。一般认为 p 值小于 0.05 或 0.01 时，结果是显著的。

表 8-1　小样本的秩表

X　值	秩
1.4	3
2.7	6
1.6	1.5
1.6	4.5
3.3	7
0.9	1
1.1	2

kruskalwallis 函数显示两个图。第 1 个图是标准方差分析表，使用数据的秩而不是值进行计算（见表 8-1）。秩通过对数据从小到大进行排序来获得，并按这个顺序设置数值索引号。

方差分析表的入口是一般平方和、自由度和其他秩的计算量。F 统计量一般用卡方统计量代替。p 值衡量卡方统计量的显著性。

第 2 个图显示 $\boldsymbol{X}$ 的每一列的箱形图。

● p = kruskalwallis(X, group) 当 X 为矩阵时，对于 X 中样本的箱形图，用 group（一个字符数组或单元数组）中的值作为标签。group 中的每一行包含 X 中对应列数据的标签，所以 group 的长度必须与 X 中列的个数相同。

当 X 为向量时，kruskalwallis 函数对 X 中的样本进行 Kruskal-Wallis 检验。group 中的每个元素确定 X 具有的对应元素的样本，所以 group 必须与 X 具有相同的长度。group 中包含的标签同样用于标注箱形图。

不必按顺序标注样本。例如，如果 X 包含在 3 个不同温度（-27℃，65℃和 110℃）上得到的观测值，则可以使用这些数值作为 group 中的样本标签。如果 group 的某行包含一个空的单元或空的字符串，则 X 中的该行和对应的观测值被忽略。输入中的空值（NaN）也同样被忽略。

● p = kruskalwallis(X, group, 'displayopt') 当'displayopt'设置为'on'时，激活表和箱形图的显示；该参数设置为'off'时，则取消显示。

● [p, table] = kruskalwallis(...) 在单元数组表中返回方差分析表（包含列和行标签）。（可以在 Edit 菜单中使用 Copy Text 选项来将方差分析表复制为文本格式。）

● [p, table, stats] = kruskalwallis(...) 返回 stats 结构，进行多元比较检验。

● Kruskal-Wallis 检验对所有样本均值相同的零假设进行评价。有时进行检验，决定哪些均值对显著差异是很可取的。通过提供 stats 结构作为输入，可以使用 multcompare 函数进行此类检验。

注意：Kruskal-Wallis 检验要求 X 中数据满足下面一些假设。

① 除了位置参数可能不同，要求所有的样本具有相同的连续分布形式；

② 所有观测值相互独立；

③ 典型方差分析对分布的要求比第一个条件更高—要求样本总体服从正态分布。

8.1.3 应用实例

【例 8-1】 用 anova1 函数对材料强度数据进行分析。3 种不同合金制成的梁的强度如下所示：

```
strength = [82  86  79  83  84  85  86  87  74  82  78  75  76  77  79  ...
            79  77  78  82  79];
alloy = {'st','st','st','st','st','st','st','st',...
         'al1','al1','al1','al1','al1','al1',...
         'al2','al2','al2','al2','al2','al2'};
```

这一次我们试着进行典型方差分析和 Kruskal-Wallis 方差分析，忽略显示：

```
anova1(strength,alloy,'off')
ans =
   1.5264e-004
kruskalwallis(strength,alloy,'off')
ans =
   0.0018
```

尽管根据 Kruskal-Wallis 检验的结果，3 种合金强度差异的显著性要稍差一些，但两种检验都发现这 3 种合金的强度具有显著差异。一种很典型的情况是，当数据集能很好地拟合正态分布时，典型方差分析对于组间差异更敏感。

为了理解什么时候使用非参数检验更合适，现在看看当数据不服从正态分布时，检验如何表现。用一个异常值替换原数据集中的一个数值（将强度数据中的第 20 个数值重新定义为 120），然后进行分析：

```
strength(20)=120;
anova1(strength,alloy,'off')
ans =
   0.2501
kruskalwallis(strength,alloy,'off')
ans =
   0.0060
```

从分析结果可以看出，典型方差分析没有发现显著差异，但非参数检验过程发现了。这就演示了非参数过程的一个特点，即对于较小的样本数据，它受异常值的影响较小。

8.2 Friedman 检验

8.2.1 基本数学原理

该法检验 k 个相关样本是否来自同一总体（是否有显著差异），进行检验时，首先建立一个 N 行 k 列的双向表，当下面的统计量近似服从自由度为 k-1 的χ^2分布时，认为秩和无显著性差异，这 k 个相关样本来自同一总体，否则否定零假设。

$$\chi^2 = \frac{12}{Nk(k+1)}\sum_{j=1}^{k}(R_j)^2 - 3N(k+1)$$

式中，N 为行数，k 为列数，R_j 为第 j 列的秩和。

8.2.2 有关函数介绍

用 friedman 函数进行二因素方差分析的 Friedman 检验，其调用格式为：

- p = friedman(X, reps) 进行非参数 Friedman 检验，比较 X 的列均值。Friedman 检验与典型二因子方差分析相似，但它在评价可能的行效应以后只对列效应进行检验。它不检验行效应和交互效应。当主要研究列代表的数据时，最好使用 Friedman 检验。

不同的列代表因子 A 的变化。不同的行代表因子 B 的变化。若对于每个因子的组合，有一个以上的观测值，则输入 reps 表示每个“单元”中的重复测量次数，它必须是常数。

下面的矩阵表示了当列因子 A 有 3 个水平，行因子 B 有两个水平，有两个重复测量次数（reps=2）时的格式。脚标分别表示行、列和重复次数。

$$\begin{vmatrix} x_{111} & x_{121} & x_{131} \\ x_{112} & x_{121} & x_{131} \\ x_{211} & x_{221} & x_{231} \\ x_{212} & x_{222} & x_{232} \end{vmatrix}$$

Friedman 检验假设模型具有下面的形式：

$$x_{ijk} = \mu + \alpha_i + \beta_j + \varepsilon_{ijk}$$

式中，μ 为总体位置参数，α_i 表示列效应，β_j 表示行效应，ε_{ijk} 表示误差。

该检验给 B 的每个水平上的数据评秩，检验 A 的不同水平。Friedman 检验返回的 p 值是零假设的 p 值。若 p 值接近于 0，则认为零假设可疑。一个足够小的 p 值说明至少有一个列样本均值与其他列样本均值之间有显著差异，即因子 A 有主效应。为了决定结果是否是"统计上显著"，需要确定 p 值。该值由用户朋友自己确定。一般地，当 p 值小于 0.05 或 0.01 时，认为结果是显著的。

Friedman 检验还生成一个显示方差分析表的图形，它将秩的变异性分为两部分或三部分：由于列均值之间存在差异而导致的变异；由于行和列之间的交互作用导致的变异（若 rteps 大于它的默认值 1）和剩下的不能解释的变异性。

方差分析表有 6 列：

第 1 列显示误差来源；

第 2 列显示源于每一个误差来源的平方和（SS）；

第 3 列为与每一个误差来源相关的自由度（df）；

第 4 列为均值平方(MS)，它是误差平方和与自由度的比值，即 SS/df；

第 5 列为 F 统计量，它是均值平方和的比值。

第 6 列为 p 值，它是 F 的函数(fcdf)。当 F 增加时 p 值减小。

- p = friedman(X,reps,'displayopt') 将'displayopt'参数设置为'on'时，激活方差分析表的显示（默认选项）；该参数设置为’off’时，则取消显示。
- [p,table] = friedman(...) 在数组表格中返回方差分析表（包含列和行标签）。
- [p,table,stats] = friedman(...) 返回 stats 结构，用于进行多元比较检验。Friedman 检验评价所有列效应相等的假设。

8.2.3 应用实例

【例 8-2】 本例重复 anova2 函数的例子，应用 Friedman 检验。重新调用爆玉米花品牌和爆玉米花方法的数据。popcorn 矩阵的列为品牌(Gourmet、National 和 Generic)，行为爆玉米花方法(Oil 和 Air)。该研究中对每一个品牌的爆玉米花用每一种方法爆了 3 次。生成的值显示在装爆玉米花的杯上。

```
load popcorn
popcorn
popcorn =
    5.5000    4.5000    3.5000
    5.5000    4.5000    4.0000
    6.0000    4.0000    3.0000
    6.5000    5.0000    4.0000
    7.0000    5.5000    5.0000
    7.0000    5.0000    4.5000
p = friedman(popcorn,3)
p =
    0.0010
```

p 值等于 0.001（很小），表示爆玉米花品牌对爆玉米花的产出有影响。该结论与 anova2 函数的结论相一致。

Friedman 检验表如图 8-1 所示。可参考描述部分的内容进行阅读。

Figure No. 1: Friedman's Test

File Edit View Insert Tools Window Help

Friedman's ANOVA Table

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Columns	99.75	2	49.875	13.76	0.001
Interaction	0.0833	2	0.0417		
Error	16.1667	12	1.3472		
Total	116	17			

Test for column effects after row effects are removed

图 8-1　Friedman 方差分析表

8.3　秩和检验

8.3.1　基本数学原理

秩和检验可检验两个总体是否相等。

设（$X_1, X_2, \dots, X_m$）是取自总体 X 的一个样本，（$Y_1, Y_2, \dots, Y_n$）是取自总体 Y 的一个样本，通过比较这两个样本的大小来比较 X 和 Y 的大小。为了进行比较，首先将两个样本合在一起：

$$(Z_1, Z_2, \dots, Z_{m+n}) = (X_1, X_2, \dots, X_m, Y_1, Y_2, \dots, Y_n),$$

称为合样本。它的秩统计量记为（$R_1, R_2, \dots, R_{m+n}$）。当 i=1, 2, …, m 时，R_i 表示 X_i 在合样本中的秩；当 i=1, 2, …, n 时，R_{m+i} 表示 Y_i 在合样本中的秩。Wilconxon 取检验统计量

$$T = \sum_{i=1}^{m} R_i$$

当零假设成立时，X 与 Y 之间没有显著差别。所以根据样本观测值得到的 T 的观测值 t 应该不会太大或太小；如果 t 太大或太小，则说明 Y 比 X 随机地小或随机地大，拒绝零假设。

8.3.2　有关函数介绍

用 ranksum 函数进行两个总体相等的 Wilcoxon 秩和检验，其调用格式为：

- p = ranksum(x, y, alpha)　假设 x 和 y 为取自相同总体的两个样本，对它们进行秩和检验，返回零假设为它们相等时的显著性概率。x 和 y 为向量，但可以具有不同的长度；若它们的长度不相等，则 x 必须比 y 小。alpha 是必要的显著性水平，且必须为 0 和 1 之间的标量。
- [p, h] = ranksum(x, y, alpha)　返回假设检验的结果 *h*。若 x 和 y 没有显著差异，则 *h* 为 0。若它们有显著差异，则 *h* 为 1。

若零假设为真，则 *p* 为使用结果与使用数据（**x** 和 **y**）的极端程度至少相当的概率。若 *p* 接近于 0，则可以拒绝零假设。

- [p, h, stats] = ranksum(x, y, alpha)　返回域值 stats.ranksum，其值等于秩和统计量。对于大样本而言，它还包含 stats.zval，用于计算 *p* 值的 normal(**Z**)统计量的值。

8.3.3　应用举例

【例 8-3】　本例检验用 poissrnd 函数创建的两个样本的均值是否相等。

```
x = poissrnd(5,10,1);
y = poissrnd(2,20,1);
[p,h] = ranksum(x,y,0.05)
```

```
p =
    0.0028
h =
    1
```

h=1, p<0.05，所以拒绝零假设，认为两个样本的均值不相等。

【例 8-4】 某医院外科用两种手术方法治疗情况基本相同的肝癌患者 11 例。患者采用随机方法分配到不同手术组。每例手术后生存月数列在下面，试比较两种手术方法的术后生存月数有否差别？

甲法　2　3　5　6　12

乙法　3　7　8　8　10　11

```
x=[2 3 5 6 12];
y=[3 7 8 8 10 11];
[p,h] = ranksum(x,y,0.05)
p =
   0.2727
h =
   0
```

h=0, p>0.05，所以接受零假设，认为两种手术方法的术后生存月数没有显著差别。

8.4　符号秩检验

8.4.1　基本数学原理

应用 Wilcoxon 符号秩检验法进行检验时，首先将所有配对数据的评分差按绝对值大小评秩，然后对每一个秩附加不同的符号，用正号表示来自正的评分差的秩，用负号表示来自负的评分差的秩。如果两个相关样本等价（没有差别），则将对应于正号的秩和对应于负号的秩分别求和以后，两个和值大致相等。若两个和值相差很大，则两个样本差异较大。

8.4.2　关函数介绍

用 signrank 函数进行中值相等的 Wilcoxon 符号秩检验，其调用格式为：

● p = signrank(x, y, alpha) 返回两个匹配样本 x 和 y 的中值相等的显著性概率。x 和 y 必须为长度相等的向量。alpha 为必要的显著性水平且必须为界于 0 和 1 之间的标量。

● [p, h] = signrank(x, y, alpha) 还返回假设检验的结果 h。若 x 和 y 的中值没有显著差异，则 h 为 0。若它们有显著差异，则 h 为 1。

若零假设为真，则 p 为使用结果与使用数据（$\boldsymbol{x}$ 和 $\boldsymbol{y}$）的极端程度至少相当的概率。p 根据反映 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 的对应元素之间的差异的秩值进行计算。若 p 接近于 0，则可以拒绝零假设。

● [p, h, stats] = signrank(x, y, alpha) 还返回 stats 结构，包含字段 stats.signedrank，其值为符号秩统计量。对于大样本，它还包含 stats.zval，用于计算 p 值的 normal（$\boldsymbol{Z}$）值。

8.4.3　应用实例

【例 8-5】 本例检验用 normrnd 函数生成的两个样本的均值是否相等。这两个样本具有相同的理论均值和不同的标准差。

```
x = normrnd(0,1,20,1);
y = normrnd(0,2,20,1);
[p,h] = signrank(x,y,0.05)
p =
    0.2568
h =
    0
```

由于 $p>0.05$，$h=0$，所以接受原假设，认为这两个样本的均值没有显著差异。

【例 8-6】 经两种处理方法处理以后的小麦，分别种在 8 对地块上，收成如下：

```
处理A   209   200   177   169   159   169   187   198
处理B   151   168   147   164   166   163   176   188
```

假设两种处理方法没有差异。

```
x=[209 200 177 169 159 169 187 198];
y=[151 168 147 164 166 163 176 188];
  [p,h]=signrank(x,y,0.05)
p =
    0.0313
h =
    1
```

$h=1$，拒绝零假设，认为两种处理方法在 0.05 的水平上存在显著差异。

8.5 符号检验

8.5.1 基本数学原理

符号检验适用于那些不适合用定量测量而能将每一对的两个成员互相分出等级的问题。进行检验时，首先对两组数据进行配对和评分。假设 N_A 和 N_B 为配对样本内样本 A 和样本 B 的评分。如果 $N_A>N_B$ 的配对数与 $N_B>N_A$ 的配对数相等，则认为两个样本间无差异。如果实际观察到的某一种配对数过少，则认为两个样本间差异显著。

8.5.2 有关函数介绍

用 signtest 函数对成对样本进行符号检验，其调用格式为：

● p = signtest(x, y, alpha) 返回两个成对样本 x 和 y 的中值相等的显著性概率。x 和 y 必须为长度相等的向量。y 也可以为一标量。在本例中，signtest 函数计算 x 的中值与常数 y 不同的概率。alpha 为必要的显著性水平且必须为一 0 和 1 之间的标量。

● [p, h] = signtest(x, y, alpha) 还返回假设检验的结果 h。若 x 和 y 的中值没有显著差异，则 h 为 0。若它们有显著差异，则 h 为 1。

若零假设为真，则 p 为使用结果与使用数据（$\boldsymbol{x}$ 和 $\boldsymbol{y}$）的极端程度至少相当的概率。p 根据反映 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 的对应元素之间的差异的秩值进行计算。若 p 接近于 0，则可以拒绝零假设。

● [p, h, stats] = signtest(x, y, alpha) 返回 stats 结构，包含域值 stats.sign，其值为符号统计量。对于大样本，它还包含 stats.zval，用于计算 p 值的 normal（$\boldsymbol{Z}$）值。

8.5.3 应用实例

【例 8-7】 本例检验用 normrnd 函数生成的两个样本是否相等。这两个样本具有相同的理论均值和不同的标准差。

```
x = normrnd(0,1,20,1);
y = normrnd(0,2,20,1);
[p,h] = signtest(x,y,0.05)
p =
   0.8238
h =
   0
```

因为 $p>0.05$, $h=0$，所以接受原假设，认为这两个样本在 0.05 的水平上没有显著差异。

【例 8-8】 在一成对实验中，用两种食物喂养的猪的增重（磅）数据如下所示

```
饲料 A   25  30  28  34  23  25  27  35  30  28  32  29  30
         30  31  29  23  26
饲料 B   19  32  21  34  19  25  25  31  31  26  30  25  29
         31  25  25  20  25
```

假设使用两种饲料养猪的效果相同。

```
x=[25 30 28 34 23 25 27 35 30 28 32 29 30 30 31 29 23 26];
y=[19 32 21 34 19 25 25 31 31 26 30 25 29 31 25 25 20 25];
[p,h] = signtest(x,y,0.05)
p =
   0.0213
h =
   1
```

$h=1$，$p<0.05$，拒绝零假设，认为两种饲料养猪的效果在 0.05 的水平上有显著差异。

第 9 章　多 元 统 计

9.1　判别分析

判别分析在生物学、医学、地质学、石油、气象等领域得到较为广泛的应用。例如地质人员需要根据化学成分等来判别采到的矿石属于哪一种矿，气象工作者需要根据采集到的数据判断近日内的天气是晴、是阴还是雨。

9.1.1　基本数学原理

判别分析是利用原有的分类信息，得到体现这种分类的函数关系式（称之为判别函数，一般是与分类相关的若干个指标的线性关系式），然后利用该函数去判断未知样品属于哪一类。因此，这是一个学习和预测的过程。

常用的判别分析方法有距离判别法、费歇尔判别法和贝叶斯法等。根据处理变量的方式的不同，又可以分为典型法和逐步法。下面分别叙述。

1．距离判别法

距离判别法有欧氏距离法和马氏距离法等。欧氏距离法比较粗糙，本软件中采用的是马氏距离法。应用马氏距离法时，首先要计算各类别的样本指标的协方差矩阵 S_i，然后利用下式计算马氏距离：

$$d_i = (x-\bar{x}^{(i)})'S_i^{-1}(x-\bar{x}^{(i)}), i=1, 2, ..., k$$

该法没有考虑样本指标值的分布。

2．费歇尔判别法

该法以费歇尔准则为标准评选判别函数。所谓费歇尔准则，指的是较优的判别函数应该能根据待判对象的 n 个指标最大限度地将它所属的类与其他类区分开来。一般应用中多采用线性判别函数。基本方法是首先假定判别函数（线性函数），然后根据已知信息对判别函数进行训练，得到函数关系式中的系数值，从而最终确定判别函数。该法有时会使误判次数增加，但由于采用线性判别函数，所以使用简便。

3．贝叶斯判别法

贝叶斯判别法是一种概率方法。它的好处是可以充分利用先验信息，可以考虑专家的意见。应用该法需要事先假定样本指标值的分布（如多元正态分布等）。贝叶斯判别函数的表达式如下所示：

$$F_i(x)=\pi_i f_i(x_1, x_2, ..., x_n), i=1, 2, ..., m$$

式中，$f_i(x_1, x_2, ..., x_n)$ 表示密度函数 f_i 在待判对象的 n 个指标值处的函数值。当

$$F_i(x)=\max F_j(x) \quad 1<j<m$$

时，认为 x 属于相应类中的任意一个。

9.1.2 有关函数介绍

1. classify 函数

用 classify 函数进行线性判别分析，其调用格式如下：

- class = classify(sample,training,group) 指定 sample 数据的每一行到训练集 training 指定的一个类中。sample 和 training 必须具有相同的列数。group 向量包含从 1 到组数的正整数，它指明训练集中的每一行属于那一个类。group 和 training 必须具有相同的行数。该函数返回 class，它是一个与 sample 具有相同行数的向量。class 的每一个元素指定 sample 中对应元素的分类。通过计算 sample 与 training 中每一行的马氏距离，Classify 函数决定 sample 中的每一个行属于哪一个分类。

【例 9-1】

```
load discrim
sample = ratings(idx,:);
training = ratings(1:200,:);
g = group(1:200);
class = classify(sample,training,g);
first5 = class(1:5)
first5 =
    2
    2
    2
    2
    2
```

2. mahal 函数

用 mahal 函数计算马氏距离，其格式如下：

- mahal(Y,X) 计算 X 矩阵中样本至 Y 矩阵中每一个点（行）的马氏距离。Y 的列数必须等于 X 的列数，但它们的行数可以不同。X 的行数必须大于列数。

马氏距离是一个源于空间中点的数据集合分割的多变量度量。在线性判别分析中，它是最小化准则。

【例 9-2】 计算矩阵 ***r*** 的马氏距离，当应用于它自身时，可以找到异常值。

```
r = mvnrnd([0 0],[1 0.9;0.9 1],100);
r = [r;10 10];
d = mahal(r,r);
last6 = d(96:101)
last6 =
   1.1036
   2.2353
   2.0219
   0.3876
   1.5571
   52.7381
```

很明显，最后一个元素是异常值。

9.1.3　应用综合实例

【例 9-3】 我国山区某大型化工厂，在厂区及邻近地区挑选有代表性的 15 个大气取样点，每日 4 次同时抽取大气样品，测定其中含有的 6 种气体的浓度，前后共 4 天，每个取样点每种气体实测 16 次。计算每个取样点每种气体的平均浓度，数据如表 9-1 所示。气体数据对应的污染地区分类如表中最后一列所示。现有两个取自该地区的 4 个气体样本，气体指标如表中后 4 行所示，试判别这 4 个样品的污染分类。

表 9-1　大气样品数据表

气体	氯	硫化氢	二氧化硫	碳 4	环氧氯丙烷	环己烷	污染分类
1	0.056	0.084	0.031	0.038	0.0081	0.022	1
2	0.040	0.055	0.100	0.110	0.0220	0.0073	1
3	0.050	0.074	0.041	0.048	0.0071	0.020	1
4	0.045	0.050	0.110	0.100	0.0250	0.0063	1
5	0.038	0.130	0.079	0.170	0.0580	0.043	2
6	0.030	0.110	0.070	0.160	0.0500	0.046	2
7	0.034	0.095	0.058	0.160	0.200	0.029	1
8	0.030	0.090	0.068	0.180	0.220	0.039	1
9	0.084	0.066	0.029	0.320	0.012	0.041	2
10	0.085	0.076	0.019	0.300	0.010	0.040	2
11	0.064	0.072	0.020	0.250	0.028	0.038	2
12	0.054	0.065	0.022	0.280	0.021	0.040	2
13	0.048	0.089	0.062	0.260	0.038	0.036	2
14	0.045	0.092	0.072	0.200	0.035	0.032	2
15	0.069	0.087	0.027	0.050	0.089	0.021	1
样品 1	0.052	0.084	0.021	0.037	0.0071	0.022	
样品 2	0.041	0.055	0.110	0.110	0.0210	0.0073	
样品 3	0.030	0.112	0.072	0.160	0.056	0.021	
样品 4	0.074	0.083	0.105	0.190	0.020	1.000	

在命令窗口输入下面的命令行:

```
training=[0.056 0.084 0.031 0.038 0.0081 0.022
          0.040 0.055 0.100 0.110 0.0220 0.0073
          0.050 0.074 0.041 0.048 0.0071 0.020
          0.045 0.050 0.110 0.100 0.0250 0.0063
          0.038 0.130 0.079 0.170 0.0580 0.043
          0.030 0.110 0.070 0.160 0.0500 0.046
          0.034 0.095 0.058 0.160 0.200  0.029
          0.030 0.090 0.068 0.180 0.220  0.039
          0.084 0.066 0.029 0.320 0.012  0.041
          0.085 0.076 0.019 0.300 0.010  0.040
          0.064 0.072 0.020 0.250 0.028  0.038
          0.054 0.065 0.022 0.280 0.021  0.040
          0.048 0.089 0.062 0.260 0.038  0.036
```

```
          0.045 0.092 0.072 0.200 0.035  0.032
          0.069 0.087 0.027 0.050 0.089  0.021];
group=[1 1 1 1 2 2 1 1 2 2 2 2 2 2 2 1]';
sample=[0.052 0.084 0.021 0.037 0.0071 0.022
         0.041 0.055 0.110 0.110 0.0210 0.0073
         0.030 0.112 0.072 0.160 0.056 0.021
         0.074 0.083 0.105 0.190 0.020 1.000];
class = classify(sample,training,group)
class =
     1
     1
     2
     2
```

所以这 4 个气体样品的污染地区分类分别为第 1 类、第 1 类、第 2 类和第 2 类。

注意：对于样本数据的每一种分类，要求其对应的观测案例数（即 training 的行数）必须大于观测的变量个数（即 training 的列数）。例如本例中，变量个数为 6，则对应于第 1 类和第 2 类的观测案例数必须大于 6，否则将出错。

9.2 系统聚类分析

在实际工作中，我们经常会遇到样品或指标的分类问题。根据事先是否已经建立类别，分类问题又可以分为判别分析和聚类分析。判别分析研究事先已经建立类别的情况，即将样品或指标按已知的类别进行归类。聚类分析则适用于事先没有分类的情况，即如何将样品或指标进行分类的问题。这里主要介绍聚类分析。聚类分析包含的内容很广泛，可以有系统聚类法、动态聚类法、分裂法、最优分割法、模糊聚类法、图论聚类法、聚类预报等多种方法。

9.2.1 基本数学原理

系统聚类法是聚类分析中应用最为广泛的一种方法，它的基本原理是：首先将一定数量的样品或指标各自看成一类，然后根据样品（或指标）的亲疏程度，将亲疏程度最高的两类进行合并。然后考虑合并后的类与其他类之间的亲疏程度，再进行合并。重复这一过程，直至将所有的样品（或指标）合并为一类。

1．距离和相似系数

衡量样品或指标之间的亲疏程度的指标有两种，即距离和相似系数。距离是将每个样品看成是 m 个变量对应的 m 维空间中的一个点，然后在该空间中所定义的，距离越近，则亲密程度越高。相似系数接近于 1 或-1 时，认为样品或指标之间的性质比较接近；相似系数接近于 0 时，认为样品或指标之间是无关的。下面是一些常用的距离和相似系数及其定义方法。

（1）欧氏距离

$$d_{ij}=\sqrt{\sum_{t=1}^{p}(x_{it}-x_{jt})^2}\quad (i,j=1,2,\ldots,n)$$

（2）标准化欧氏距离

$$d_{rs}^2=(x_r-x_s)D^{-1}(x_r-x_s)'$$

（3）马氏距离

$$d_{rs}^2=(x_r-x_s)'V^{-1}(x_r-x_s)'$$

（4）布洛克距离

$$d_{rs}=\sum_{j=1}^{n}\left|x_{rj}-x_{sj}\right|$$

（5）切比雪夫距离

$$d_{ij}(\infty)=\max|x_{ik}-x_{jk}|\quad(i,j=1,2,\ldots,n)$$

（6）明可夫斯基距离

$$d_{rs}=\left\{\sum_{j=1}^{n}\left|x_{rj}-x_{sj}\right|^{p}\right\}^{1/p}$$

注意，当 p=1 时，即为布洛克距离；当 p=2 时，为欧氏距离。

（7）夹角余弦(相似系数)

$$\cos\alpha_{ij}=\frac{\sum_{k=1}^{m}x_{ki}\cdot x_{kj}}{\sqrt{(\sum_{k=1}^{n}x_{ki}^2)(\sum_{k=1}^{n}x_{kj}^2)}}$$

（8）相关系数(相似系数)

$$r_{ij}=\frac{\sum_{k=1}^{n}(x_{ki}-\bar{x}_i)(x_{kj}-\bar{x}_j)}{\{[\sum_{k=1}^{n}(x_{ki}-\bar{x}_i)^2]\cdot[\sum_{k=1}^{n}(x_{kj}-\bar{x}_j)^2]\}^{1/2}}$$

2．常用的聚类方法

常用的聚类方法主要有以下几种。

（1）最短距离法

该算法将两个类之间的距离定义为一个类的所有个体与另一个类的所有个体之间的距离的最小者，即

$$D_{pq}=\min d_{ij}\quad x_i\in G_p,x_j\in G_q$$

（2）最长距离法

与最短距离法相反，该法用个体之间的最远距离来定义类与类之间的距离，即

$$D_{ij}=\max\{d_{Kl}\}\quad x_k\in G_i,x_l=G_j$$

（3）中间距离法

该法在定义类与类之间的距离时，采用的是最短距离与最长距离之间的中间距离。当用两类 G_p 与 G_q 合并成新类 G_r 时，任一类 G_i 与 G_r 之间的距离用下式计算。

$$D_{ir}=\sqrt{\frac{1}{2}D_{ip}^2+\frac{1}{2}D_{iq}^2-\frac{1}{4}D_{pq}^2}$$

式中，D_{ip}, D_{iq}, D_{pq} 分别为 G_i, G_p, G_q 之间的距离。

（4）重心法

该法将两类之间的距离定义为两类重心之间的距离。它考虑了每一类所包含的样品数，每一类重心即为该类样品的均值。

假设类 G_p 和 G_q 合并成 G_r 以后，它们的样本数目分别是 n_p, n_q 和 $n_r=n_p+n_q$。则 G_r 与其他类 G_i 之间的距离为

$$D_{ir}^2 = \frac{n_p}{n_r} D_{ip}^2 + \frac{n_q}{n_r} D_{iq}^2 - \frac{n_p}{n_r}\frac{n_q}{n_r} D_{pq}^2$$

（5）离差平方和法

该法是 Ward 根据方差分析的原理得到的。如果分类比较合理，则同类样品之间的离差平方和较小，类与类之间的离差平方和较大。假设类 G_p 与类 G_q 合并成新类 G_r，则 G_r 与任一类 G_i 的距离递推公式为

$$D_{ir}^2 = \frac{n_i + n_p}{n_r + n_i} D_{ip}^2 + \frac{n_i + n_q}{n_r + n_i} D_{iq}^2 - \frac{n_i}{n_r + n_i} D_{pq}^2$$

利用离差平方和分类的效果比较好，它要求样品之间的距离必须是欧氏距离。

（6）平均联结法

前面介绍了用类之间的最小距离、最大距离和中间距离等联结类的方法，还可以用取平均的方法联结类。平均联结法分为两种，即组间平均联结法和组内平均联结法。

组间平均联结法将两个类所有成对案例（各来自一个类）间的平均距离作为类间距离并要求该距离最小。它利用了两个类中所有成对案例的信息。

组内平均联结法的目的则是要使产生类的所有个案之间的平均距离尽可能地小。

3．数据的转换

进行聚类分析时，各变量之间有可能存在不同量纲、不同数量级的情况，因此存在转换数据的必要性。转换数据的目的是使这些变量具有可比性。常用的数据转换方法有中心化变换、极差正规化和标准化等。

9.2.2 有关函数介绍

1．pdist 函数

用 pdist 函数计算观测量之间的匹配距离，其调用格式为：

● Y = pdist(X) 计算 X 矩阵中配对样本的欧氏距离。X 为一 $m\times n$ 的矩阵，可以看做大小为 n 的 m 个向量。对于由 m 个样本组成的数据集合，将有$(m-1)\times m/2$ 个匹配对。Y 为长度$(m-1)\times m/2$ 的向量，包含距离信息。这些距离信息按照(1, 2), (1, 3), ..., (1, m), (2, 3), ..., (2, m), ..., ...,(m-1, m) 的顺序排列。Y 也常称为相似矩阵或不相似矩阵。

为了节省空间和计算时间，Y 用向量的格式保存。但可以用 squareform 函数将它转化为平方矩阵，这样，矩阵中的元素(i,j)对应于原始数据中对象 i 与 j 之间的距离。

● Y = pdist(X,'metric') 使用'metric'指定的方法计算 X 数据矩阵中对象之间的距离。'metric'可以是下面字符串中的任意一个：

'Euclid'——欧氏距离（默认选项）；

'SEuclid'——标准化欧氏距离；

'Mahal'——马氏距离；

'CityBlock'——布洛克距离；

'Minkowski'——明可夫斯基距离。

● Y = pdist(X,'minkowski', p) 使用明可夫斯基距离计算 X 数据矩阵中对象之间的距离。p

为明可夫斯基距离计算过程中的幂次，默认值为 2。

【例 9-4】

```
X = [1 2; 1 3; 2 2; 3 1]
X =
     1     2
     1     3
     2     2
     3     1
Y = pdist(X,'mahal')
Y =
    2.3452    2.0000    2.3452    1.2247    2.4495    1.2247
Y = pdist(X)
Y =
    1.0000    1.0000    2.2361    1.4142    2.8284    1.4142
squareform(Y)
ans =
         0    1.0000    1.0000    2.2361
    1.0000         0    1.4142    2.8284
    1.0000    1.4142         0    1.4142
    2.2361    2.8284    1.4142         0
```

2．squareform 函数

该函数将 pdist 函数的输出重定义为平方矩阵的格式。调用格式为：

● S = squareform(Y) 将 pdist 函数返回的距离信息 Y 重新定义为平方矩阵的格式。该格式中，$S(i, j)$ 表示原始数据中 i 观测量和 j 观测量之间的距离。

3．Linkage 函数

利用该函数创建系统聚类树。调用格式为：

● Z = linkage(Y) 使用最短距离法创建一个系统聚类树。输入矩阵 Y 为 pdist 函数的输出，是一距离向量，长度为$(m-1)\times m/2\times 1$，其中 m 为原始数据集合中的对象数。

● Z = linkage(Y,'method') 用'method'参数指定的算法计算系统聚类树。'method'可以有下面一些取值：

'single'——最短距离法（默认选项）；

'complete'——最长距离法；

'average'——平均距离法；

'centroid'——重心距离法；

'ward'——平方和递增法。

输出 $\boldsymbol{Z}$ 为一包含聚类树信息的（m-1）×3 的矩阵。聚类系统的叶节点（没有次级节点的节点）为原始数据集合中的对象，从 1 到 m 编号。这些叶节点构成了聚类树的基础。基于这些叶节点，可以建成更高的类。每一个新生成的类对应于 $\boldsymbol{Z}$ 中的第 i 行，并指定指数 $m+i$，其中 m 为初始叶节点的总个数。

列 1 和列 2，即 $\boldsymbol{Z}(i, 1{:}2)$，包含组成新类的配对对象的指数。该新类指定指数值 $m+i$。有 $m-i$ 个更高的类对应于系统聚类树的内节点。

第 3 列，即 $Z(i, 3)$，包含每一个第 i 行类中配对对象之间对应的联结距离。

例如，考虑一个有 30 个初始节点的个案。若由联结函数组成的第十个类连接对象 5 和对象 7，并且它们的距离是 1.5，则 **Z** 的第 10 行将包含值(5, 7, 1.5)。该新组成的类将有指数 10+30=40。若类 40 在后面的行中显示了，则意味着新构成的类又被合并到某个更高的类中去了。

【例 9-5】

```
X = [3 1.7; 1 1; 2 3; 2 2.5; 1.2 1; 1.1 1.5; 3 1];
Y = pdist(x);
Z = linkage(y)
Z =
    2.0000   5.0000   0.2000
    3.0000   4.0000   0.5000
    8.0000   6.0000   0.5099
    1.0000   7.0000   0.7000
   11.0000   9.0000   1.2806
   12.0000  10.0000   1.3454
```

4. dendrogram 函数

该函数输出冰柱图，其调用格式为：

● H = dendrogram(Z) 生成系统聚类树 Z 的冰柱图。Z 是一个（m-1）×3 的矩阵，由 linkage 函数生成，其中 m 是原始数据集合中的对象个数。

冰柱图是由许多倒置的 U 形线连接系统聚类树中的对象绘成。除了 Ward 连接，每一个 U 的高度代表两个相连对象之间的距离。输出 H 是一个表示线的句柄的向量。

● H = dendrogram(Z, p) 生成只有顶部 p 个节点的冰柱图。默认时，dendrogram 函数使用 30 作为 p 的值。当有 30 个以上的初始节点时，冰柱图将显得很拥挤。设置 p=0，显示所有节点。

● [H, T] = dendrogram(...) 创建一个冰柱图（见图 9-1），返回一个大小为 m 的向量 T，其中包含了原始数据集合中每一个对象的聚类个数。T 提供了到达没有在聚类树中显示的节点（因为它们位于 p 的阈值以下）的途径。例如，为了找到哪个对象包含在冰柱图的叶节点 k 中，使用 find(T==k)函数进行查找。叶节点为冰柱图底部的节点，它们下面再没有其他节点。

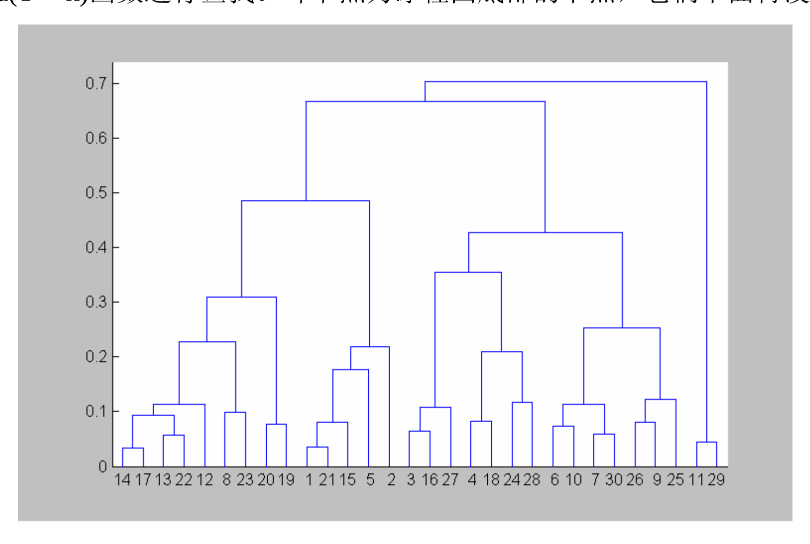


图 9-1　冰柱图

当原始数据中的对象少于 p 个时，所有的对象都会显示在冰柱图中。

【例 9-6】

```
rand('seed',12);
X= rand(100,2);
Y= pdist(X,'citiblock');
Z= linkage(Y,'average');
[H, T] = dendrogram(Z);
find(T==20)
ans =
     20
     49
     62
     65
     73
     96
```

该输出显示冰柱图中的第 20 个叶节点包含原始数据点 20, 49, 62, 65, 73 和 96。

5. cophenet 函数

利用该函数计算 Cophenetic 相关系数。其调用格式如下：

● c = cophenet(Z, Y) 计算 cophenetic 相关系数。该系数比较由 linkage 函数生成 Z 中的距离信息和由 pdist 函数生成的 Y 中的距离信息。Z 是一个$(m-1)\times 3$ 的矩阵，距离信息在第 3 列中，Y 为一大小为$(m-1)\times m/2$ 的向量。

例如，给定一距离为 Y 的一组对象$\{1,2,....m\}$ ，linkage 函数生成一个系统聚类树。cophenet 函数衡量该分类的歪曲程度，表示数据与该分类结构的拟合程度。

输出值 c 为 cophenet 相关系数。该值的大小越接近于 1 越好。该度量可以用于比较用不同算法得到的分类解。

$Z(:,3)$与 Y 之间的 cophenetic 相关系数定义为：

$$c=\frac{\sum_{i<j}(Y_{ij}-y)(Z_{ij}-Z)}{\sqrt{\sum_{i<j}(Y_{ij}-y)^2\sum_{i<j}(Z_{ij}-z)^2}}$$

式中：Y_{ij} 为 Y 中对象 i 和对象 j 之间的距离；Z_{ij} 为 $Z(:,3)$中对象 i 和对象 j 之间的距离；y 和 z 分别为 Y 和 $Z(:,3)$的平均值。

【例 9-7】

```
rand('seed',12);
X = [rand(10,3);rand(10,3)+1;rand(10,3)+2];
Y = pdist(X);
Z = linkage(Y,'centroid');
c = cophenet(Z,Y)
c =
   0.6985
```

6. cluster 函数

使用 cluster 函数，根据 linkage 函数的输出创建聚类，其调用格式如下：

● cluster(Z, cutoff) 根据 linkage 函数生成的系统聚类树 Z 来创建聚类。Z 是一个$(m-1)\times 3$ 矩阵。其中 m 为原始数据中观测量的个数。cutoff 是一个临界值，它决定 cluster 函数怎样

创建聚类。cutoff 值决定 cluster 函数怎样去解释它。

当 0 < cutoff < 2 时，cutoff 可以解释为非连续系数的阈值。非连续系数表示系统聚类树中对象之间的差异程度。若某联结的不连续系数大于阈值，则 cluster 函数将此联结作为聚类组的边界。

当 cutoff >= 2 时，cutoff 可解释为聚类的最大个数。

● cluster(Z, cutoff, depth) 创建源于聚类树 *Z* 的聚类。depth 变量指定系统聚类树的水平数，并包含在不连续系数的计算中。当指定 depth 变量以后，cutoff 变量将一直被解释为不连续系数的阀值。

flag 变量覆盖 cutoff 变量的默认意义。若 flag 为'inconsistent', 则对于不连续系数而言，cutoff 可以解释为门槛值。若 flag 为'clusters', 则 cutoff 是最大分类数。

输出 ***T*** 是一个大小为 *m* 的向量，其大小与每一对象的分类数相等。用 find(T==i)来找出哪一个源于原始数据集合的对象包含在第 *i* 个类别中。

【例 9-8】 用 pdist 函数计算随机数矩阵中项目之间的距离，然后用 linkage 函数计算基于矩阵的聚类树。linkage 函数的输出传给 cluster 函数。cutoff 值表示希望将所有项目分为 3 类。本例用 find 函数列出所有归为第 2 类的项目：

```
rand('seed', 0);
X = [rand(10,3); rand(10,3)+1; rand(10,3)+2];
Y = pdist(X);
Z = linkage(Y);
T = cluster(Z,3);
find(T == 3)
ans =
    11
    12
    13
    14
    15
    16
    17
    18
    19
    20
```

7. clusterdata 函数

利用该函数，根据数据创建分类，其调用格式如下：

● T = clusterdata(X, cutoff)根据数据矩阵 X 创建分类。X 为一 $m\times n$ 的矩阵，*m* 为观测量个数，*n* 为变量个数。cutoff 为一阈值，决定 cluster 函数如何创建分类，cutoff 的值决定 clusterdata 如何解释它。

当 0 < cutoff < 1 时，cutoff 可以解释为不连续系数的阈值，不连续系数为系统聚类树中对象的差异程度，若某联结的不连续系数大于阈值，则 cluster 函数用该联结作为分类的界限。

当 cutoff >= 1 时，cutoff 解释为系统聚类树中分类的最大个数。

输出 ***T*** 是一个大小为 *m* 的向量，其大小与每一对象的分类数相等。

● T = clusterdata(X, cutoff)与下面的命令行意义相同：

```
Y = pdist(X,'euclid');
Z = linkage(Y,'single');
T = cluster(Z,cutoff);
```

【例 9-9】 本例首先创建一个随机数样本数据集合，然后用 clusterdata 函数计算数据集合中不同项目之间的距离，并创建一个源于数据集的系统聚类树。最后，clusterdata 函数将数据集合中的项目分为 3 类。本例用 find 函数列出所有归为第 2 类的项目。

```
rand('seed', 12);
X = [rand(10,3); rand(10,3)+1.2; rand(10,3)+2.5;
T = clusterdata(X,3);
find(T == 2)
ans =
    21
    22
    23
    24
    25
    26
    27
    28
    29
    30
```

8．inconsistent 函数

该函数计算聚类树的不连续系数，其调用格式如下：

- Y = inconsistent(Z) 对于系统聚类树 Z 的每个联结计算不连续系数，其中 Z 为$(m-1)\times 3$的矩阵，由 linkage 函数生成。不连续系数通过对聚类树中每个联结的长度与系统聚类树上同一水平的其他联结的平均长度进行比较来识别联结。该系数的值越大，则由对应联结联系的对象的相似性越差。
- Y = inconsistent(Z,d) 计算系统聚类树 Z 中可每个联结至深度 d 的不连续系数，其中，d 为整数，表示计算中包含的聚类树的水平数。默认时，d=2。输出 Y 为$(m-1)\times 4$的矩阵，具有表 9-2 示出的格式。

表 9-2　***Y*** 矩阵各列的描述

列	描　述
1	计算中包含的所有联结长度的均值
2	计算中包含的所有联结的标准离差
3	计算中包含的联结个数
4	不连续系数

对于每个联结 k，不连续系数的计算为

$$\boldsymbol{Y}(k,4)=(\boldsymbol{z}(k,3)-\boldsymbol{Y}(k,1))/\boldsymbol{Y}(k,2)$$

对于叶节点（没有次级节点的节点），不连续系数设置为 0。

【例 9-10】

```
rand('seed',12);
X = rand(10,2);
Y = pdist(X);
Z = linkage(Y,'centroid');
W = inconsistent(Z,3)
W =
    0.0423  0       1.0000  0
    0.1406  0       1.0000  0
    0.1163  0.1047  2.0000  0.7071
    0.2101  0       1.0000  0
    0.2054  0.0886  3.0000  0.6792
    0.1742  0.1762  3.0000  0.6568
    0.2336  0.1317  4.0000  0.6408
    0.3081  0.2109  5.0000  0.7989
    0.4610  0.3728  4.0000  0.8004
```

9.2.3 应用综合实例

【例 9-11】 为了研究世界各国森林、草原资源的分布规律，共抽取了 21 个国家的数据，每个国家 4 项指标，原始数据见表 9-3。试用该数据对国别进行聚类分析。

表 9-3 原始数据表

国 别	森林面积（万公顷)	森林覆盖率(%)	林木蓄积量(亿立方米)	草原面积(万公顷)
中 国	11978	12.5	93.5	31908
美 国	28446	30.4	202.0	23754
日 本	2501	67.2	24.8	58
德 国	1028	28.4	14.0	599
英 国	210	8.6	1.5	1147
法 国	1458	26.7	16.0	1288
意大利	635	21.1	3.6	514
加拿大	32613	32.7	192.8	2385
澳大利亚	10700	13.9	10.5	45190
前苏联	92000	41.1	841.5	37370
捷 克	458	35.8	8.9	168
波 兰	868	27.8	11.4	405
匈牙利	161	17.4	2.5	129
南斯拉夫	929	36.3	11.4	640
罗马尼亚	634	26.7	11.3	447
保加利亚	385	34.7	2.5	200
印 度	6748	20.5	29.0	1200
印度尼西亚	2180	84.0	33.7	1200
尼日利亚	1490	16.1	0.8	2090
墨西哥	4850	24.6	32.6	7450
巴 西	57500	67.6	238.0	15900

MATLAB 提供了两种方法进行聚类分析。

一种是一次聚类。它的优点是可利用 clusterdata 函数对样本数据进行一次聚类。其缺点是可供用户选择的面比较窄，不能更改距离的计算方法。

另一种是分步聚类。可以分以下步骤进行分步聚类：

① 找到数据集合中变量两两之间的相似性和非相似性，用 pdist 函数计算变量之间的距离；

② 用 linkage 函数定义变量之间的连接；

③ 用 cophenetic 函数评价聚类信息；

④ 用 cluster 函数创建聚类。

下面分别介绍这两种聚类方法。

1．一次聚类

在命令窗口输入下面的命令行：

```
X=[11978   12.5    93.5    31908
   28446   30.4   202.0    23754
    2501   67.2    24.8       58
    1028   28.4    14.0      599
     210    8.6     1.5     1147
    1458   26.7    16.0     1288
     635   21.1     3.6      514
   32613   32.7   192.8     2385
   10700   13.9    10.5    45190
   92000   41.1   841.5    37370
     458   35.8     8.9      168
     868   27.8    11.4      405
     161   17.4     2.5      129
     929   36.3    11.4      640
     634   26.7    11.3      447
     385   34.7     2.5      200
    6748   20.5    29.0     1200
    2180   84.0    33.7     1200
    1490   16.1     0.8     2090
    4850   24.6    32.6     7450
   57500   67.6   238.0    15900];
T = clusterdata(X,0.9)
T =
    1
    1
    2
    3
    5
    6
    3
    1
    1
   10
    4
```

```
3
4
3
3
4
7
6
6
8
9
```

可见 MATLAB 将数据集合分为 10 类。调整 cutoff 值，将有不同的分类。

2．分步聚类

（1）寻找变量之间的相似性

用 pdist 函数计算相似性矩阵或非相似性矩阵。有多种方法可以计算距离。默认时用 pdist 函数计算欧氏距离，可以指定一种或多种选项（参见函数部分的内容）。进行计算之前一般最好先将数据（用 zscore 函数）进行标准化。下面的代码返回变量之间的距离信息。

```
Y=pdist(X)
Y =
  1.0e+004*
  Columns  1  through 7
    1.8376    3.3230    3.3169    3.2935    3.2377    3.3380    3.6020
  Columns  8  through 14
    1.3344    8.0212    3.3766    3.3405    3.3905    3.3163    3.3444
  Columns  15  through 21
    3.3761    3.1150    3.2233    3.1609    2.5476    4.8255    3.5138
  Columns  22  through 28
    3.5888    3.6172    3.5116    3.6243    2.1771    2.7829    6.4999
  Columns  29  through 35
    3.6601    3.6135    3.6854    3.5937    3.6287    3.6637    3.1297
  Columns  36  through 42
    3.4621    3.4583    2.8681    3.0097    0.1570    0.2537    0.1613
  Columns  43  through 49
    0.1922    3.0202    4.5871    9.6969    0.2046    0.1670    0.2342
  Columns  50  through 56
    0.1677    0.1908    0.2121    0.4398    0.1186    0.2270    0.7756
  Columns  57  through 63
    5.7236    0.0985    0.0812    0.0402    3.1636    4.5628    9.8126
  Columns  64  through 70
    0.0715    0.0251    0.0986    0.0107    0.0422    0.0757    0.5752
  Columns  71  through 77
    0.1301    0.1561    0.7845    5.8509    0.1256    0.0763    3.2427
  Columns  78  through 84
    4.5275    9.8682    0.1010    0.0992    0.1019    0.0880    0.0819
  Columns  85  through 91
    0.0963    0.6538    0.1972    0.1590    0.7827    5.9160    0.1130
  Columns  92  through 98
```

```
  3.1175   4.4864    9.7470   0.1502   0.1062   0.1739   0.0837
 Columns  99  through 105
  0.1177   0.1528    0.5291   0.0730   0.0803   0.7034   5.7916
 Columns  106  through  112
  3.2033   4.5796    9.8522   0.0389   0.0257   0.0611   0.0320
 Columns  113  through 119
  0.0068   0.0402    0.6151   0.1692   0.1793   0.8116   5.8910
 Columns  120  through  126
  4.8088   6.8929    3.2232   3.1807   3.2531   3.1733   3.2038
 Columns  127  through  133
  3.2303   2.5893    3.0457   3.1125   2.8222   2.8320   8.1679
 Columns  134  through  140
  4.6172   4.5852    4.6277   4.5609   4.5861   4.6157   4.4167
 Columns  141  through  147
  4.4808   4.4073    3.8191   5.5210   9.8816   9.8347   9.9106
 Columns  148  through  154
  9.8202   9.8548    9.8872   9.2611   9.6833   9.7147   9.2147
 Columns  155  through  161
  4.0640   0.0474    0.0300   0.0667   0.0330   0.0080   0.6374
 Columns  162  through  168
  0.2008   0.2182    0.8504   5.9172   0.0759   0.0243   0.0238
 Columns  169  through  175
  0.0525   0.5934    0.1535   0.1796   0.8093   5.8714   0.0923
 Columns  176  through  182
  0.0570   0.0236    0.6674   0.2287   0.2369   0.8694   5.9469
 Columns  183  through  189
  0.0353   0.0700    0.5846   0.1372   0.1555   0.7858   5.8593
 Columns  190  through  196
  0.0351   0.6160    0.1721   0.1853   0.8174   5.8929   0.6441
 Columns  197  through  203
  0.2056   0.2189    0.8515   5.9234   0.4568   0.5333   0.6532
 Columns  204  through  210
  5.2838   0.1129    0.6797   5.7240   0.6326   5.7688   5.3324
```

为了便于阅读，可以用 squareform 函数将距离向量转化为矩阵：

```
squareform(Y)
ans =
1.0e+004*
Columns 1 through 11
0 1.8376 3.3230 3.3169 3.2935 3.2377 3.3380 3.6020 1.3344 8.0212 3.3766
1.8376 0 3.5138 3.5888 3.6172 3.5116 3.6243 2.1771 2.7829 6.4999 3.6601
3.3230 3.5138 0 0.1570 0.2537 0.1613 0.1922 3.0202 4.5871 9.6969 0.2046
3.3169 3.5888 0.1570 0 0.0985 0.0812 0.0402 3.1636 4.5628 9.8126 0.0715
3.2935 3.6172 0.2537 0.0985 0 0.1256 0.0763 3.2427 4.5275 9.8682 0.1010
3.2377 3.5116 0.1613 0.0812 0.1256 0 0.1130 3.1175 4.4864 9.7470 0.1502
3.3380 3.6243 0.1922 0.0402 0.0763 0.1130 0 3.2033 4.5796 9.8522 0.0389
3.6020 2.1771 3.0202 3.1636 3.2427 3.1175 3.2033 0 4.8088 6.8929 3.2232
1.3344 2.7829 4.5871 4.5628 4.5275 4.4864 4.5796 4.8088 0 8.1679 4.6172
8.0212 6.4999 9.6969 9.8126 9.8682 9.7470 9.8522 6.8929 8.1679 0 9.8816
```

```
3.3766 3.6601 0.2046 0.0715 0.1010 0.1502 0.0389 3.2232 4.6172 9.8816 0
3.3405 3.6135 0.1670 0.0251 0.0992 0.1062 0.0257 3.1807 4.5852 9.8347 0.0474
3.3905 3.6854 0.2342 0.0986 0.1019 0.1739 0.0611 3.2531 4.6277 9.9106 0.0300
3.3163 3.5937 0.1677 0.0107 0.0880 0.0837 0.0320 3.1733 4.5609 9.8202 0.0667
3.3444 3.6287 0.1908 0.0422 0.0819 0.1177 0.0068 3.2038 4.5861 9.8548 0.0330
3.3761 3.6637 0.2121 0.0757 0.0963 0.1528 0.0402 3.2303 4.6157 9.8872 0.0080
3.1150 3.1297 0.4398 0.5752 0.6538 0.5291 0.6151 2.5893 4.4167 9.2611 0.6374
3.2233 3.4621 0.1186 0.1301 0.1972 0.0730 0.1692 3.0457 4.4808 9.6833 0.2008
3.1609 3.4583 0.2270 0.1561 0.1590 0.0803 0.1793 3.1125 4.4073 9.7147 0.2182
2.5476 2.8681 0.7756 0.7845 0.7827 0.7034 0.8116 2.8222 3.8191 9.2147 0.8504
4.8255 3.0097 5.7236 5.8509 5.9160 5.7916 5.8910 2.8320 5.5210 4.0640 5.9172
Columns 12 through 21
3.3405 3.3905 3.3163 3.3444 3.3761 3.1150 3.2233 3.1609 2.5476 4.8255
3.6135 3.6854 3.5937 3.6287 3.6637 3.1297 3.4621 3.4583 2.8681 3.0097
0.1670 0.2342 0.1677 0.1908 0.2121 0.4398 0.1186 0.2270 0.7756 5.7236
0.0251 0.0986 0.0107 0.0422 0.0757 0.5752 0.1301 0.1561 0.7845 5.8509
0.0992 0.1019 0.0880 0.0819 0.0963 0.6538 0.1972 0.1590 0.7827 5.9160
0.1062 0.1739 0.0837 0.1177 0.1528 0.5291 0.0730 0.0803 0.7034 5.7916
0.0257 0.0611 0.0320 0.0068 0.0402 0.6151 0.1692 0.1793 0.8116 5.89101
3.1807 3.2531 3.1733 3.2038 3.2303 2.5893 3.0457 3.1125 2.8222 2.8320
4.5852 4.6277 4.5609 4.5861 4.6157 4.4167 4.4808 4.4073 3.8191 5.5210
9.8347 9.9106 9.8202 9.8548 9.8872 9.2611 9.6833 9.7147 9.2147 4.0640
0.0474 0.0300 0.0667 0.0330 0.0080 0.6374 0.2008 0.2182 0.8504 5.9172
0 0.0759 0.0243 0.0238 0.0525 0.5934 0.1535 0.1796 0.8093 5.8714
0.0759 0 0.0923 0.0570 0.0236 0.6674 0.2287 0.2369 0.8694 5.9469
0.0243 0.0923 0 0.0353 0.0700 0.5846 0.1372 0.1555 0.7858 5.8593
0.0238 0.0570 0.0353 0 0.0351 0.6160 0.1721 0.1853 0.8174 5.8929
0.0525 0.0236 0.0700 0.0351 0 0.6441 0.2056 0.2189 0.8515 5.9234
0.5934 0.6674 0.5846 0.6160 0.6441 0 0.4568 0.5333 0.6532 5.2838
0.1535 0.2287 0.1372 0.1721 0.2056 0.4568 0 0.1129 0.6797 5.7240
0.1796 0.2369 0.1555 0.1853 0.2189 0.5333 0.1129 0 0.6326 5.7688
0.8093 0.8694 0.7858 0.8174 0.8515 0.6532 0.6797 0.6326 0 5.3324
5.8714 5.9469 5.8593 5.8929 5.9234 5.2838 5.7240 5.7688 5.3324 0
```

（2）定义变量之间的连接

数据中变量之间的近似性计算出来以后，使用 linkage 函数就可以决定数据集合中哪一些变量可以归入到某类中。该函数利用 pdist 函数生成的距离信息，连接相近的成对变量形成二分类。linkage 函数然后将新生成的类与其他变量相连接生成更大的类，直到原始数据中所有的变量被连接到系统聚类树中。

```
Z=linkage(Y)
Z =
  1.0e+004*
    0.0007    0.0015    0.0068
    0.0011    0.0016    0.0080
    0.0004    0.0014    0.0107
    0.0023    0.0013    0.0236
    0.0022    0.0012    0.0238
    0.0024    0.0026    0.0243
```

```
    0.0027    0.0025    0.0330
    0.0006    0.0018    0.0730
    0.0028    0.0005    0.0763
    0.0029    0.0019    0.0803
    0.0030    0.0031    0.0812
    0.0003    0.0032    0.1186
    0.0033    0.0017    0.4398
    0.0034    0.0020    0.6326
    0.0001    0.0009    1.3344
    0.0036    0.0002    1.8376
    0.0037    0.0008    2.1771
    0.0038    0.0035    2.5476
    0.0039    0.0021    2.8320
    0.0040    0.0010    4.0640
```

输出信息中，每 1 行代表一次连接，前两列代表连接的变量，第 3 列代表连接对象之间的距离。例如，首先国家 7 和国家 15 相连接，形成新类 22，它们之间的距离是 68；然后国家 16 和国家 11 相连，形成新类 23，它们之间的距离为 80，比前一组略大；接着，国家 4 和国家 14 相连，形成新类 24；第 4 步则是新类 23 与国家 13 相连，形成新类 25；依次类推，最后新类 40 和国家 10 相连，形成整个系统。整个输出信息展示了聚类的过程。

（3）评价聚类信息

连接变量生成聚类树以后，可能想修改聚类树或希望了解变量连接的更多信息。下面分别予以介绍。

① 修改聚类树：衡量通过 linkege 函数生成的聚类信息的有效性的一种方法是将它与 pdist 函数生成的原始相似数据对比。如果有效，则二者强相关。cophenet 函数比较这两组值，并且计算它们的相关性，返回 cophenetic 相关系数。该值越接近于 1，表示聚类效果越好。

可以通过 cophenetic 相关系数来比较用不同距离计算方法或聚类算法对同一套数据得到的聚类结果。对于本例：

```
C=cophenet(Z,Y)
C =
    0.9393
```

将 pdist 函数中距离的计算方法分别指定为“Mahal”，“SEuclid”和“CityBlock”，重新计算 pdist 函数以后，用 cophenet 函数计算，得 C 值分别等于 0.7486、0.8018 和 0.9291。均小于距离为欧氏距离时的计算结果，所以使用默认设置时的效果最佳。

② 了解与聚类连接相关的更多信息：决定数据集合中聚类的方法之一是比较聚类树中每一连接的长度与相邻次一级连接的长度。如果二者相近，则表示在此水平上变量之间是相似的，这些连接被认为具有较高水平的连续性；如果二者不相近，则表示聚类树中变量之间不相似，称为该连接与周围是不连续的。cluster 函数用不连续性的度量决定在哪里对数据进行分类。

下面的命令行生成聚类树：

```
dendrogram(Z)
ans =
   99.0009
  101.0009
```

```
102.0004
103.0004
104.0004
105.0002
106.0002
107.0002
108.0002
109.0002
110.0002
111.0002
112.0002
113.0002
114.0002
115.0002
116.0002
117.0002
118.0002
119.0002
```

图 9-2 中纵轴为变量之间的距离，横轴为聚类变量。聚类树从左到右反映了聚类的先后次序。该图可以与 linkage 函数的计算结果相对应。

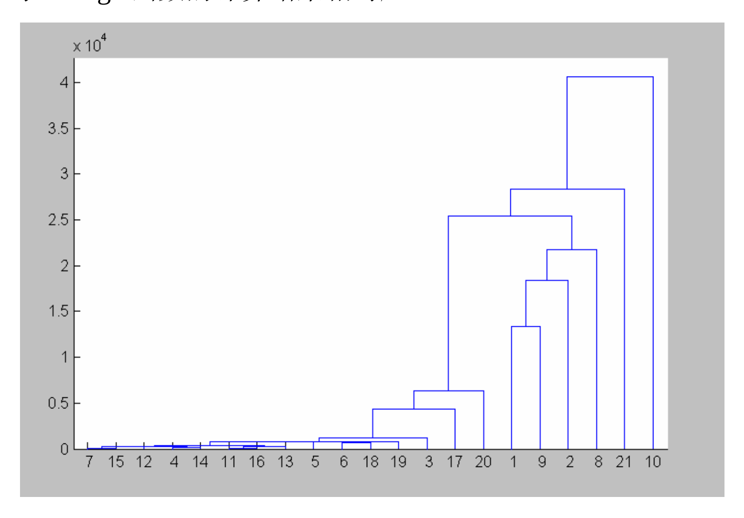


图 9-2 本问题的冰柱图

系统聚类树中每一个连接的相对连续性可用不连续性系数来定量表达，该函数比较某连接的长度与相邻连接长度的均值。若该变量与周围变量连续，则不连续性系数较低。反之则反。

用 inconsistent 函数生成聚类树每一连接的不连续系数表。该函数比较每一连接与以下两个水平的连接，称为比较的深度。用该函数可以指定其他深度。聚类树底部的对象称为叶节点，不连续系数为 0。

```
I=inconsistent(Z)
I =
  1.0e+004*
    0.0068        0   0.0001        0
```

```
    0.0080         0    0.0001         0
    0.0107         0    0.0001         0
    0.0158    0.0110    0.0002    0.0001
    0.0153    0.0120    0.0002    0.0001
    0.0196    0.0077    0.0003    0.0001
    0.0270    0.0053    0.0003    0.0001
    0.0730         0    0.0001         0
    0.0546    0.0306    0.0002    0.0001
    0.0766    0.0052    0.0002    0.0001
    0.0793    0.0026    0.0003    0.0001
    0.0999    0.0265    0.0002    0.0001
    0.2792    0.2271    0.0002    0.0001
    0.5362    0.1363    0.0002    0.0001
    1.3344         0    0.0001         0
    1.5860    0.3559    0.0002    0.0001
    2.0074    0.2401    0.0002    0.0001
    1.7858    1.0157    0.0003    0.0001
    2.6898    0.2011    0.0002    0.0001
    3.4480    0.8711    0.0002    0.0001
```

矩阵中，第 1 列为所有连接长度的均值，第 2 列为所有连接长度的标准差，第 3 列为计算中所包含的连接数，第 4 列为不连续系数。该输出信息可以与 linkage 函数的输出对照阅读，第 1 行代表国家 7 和国家 15 相连，因为二者均为叶节点，所以不连续系数为 0。

（4）创建聚类

创建二分聚类树以后，可以用 cluster 函数将系统分成更大的类。有两种方式创建聚类：

① 找到原始数据集合的自然分界：如果用 cluster 函数对数据进行分类，指定一个不连续系数 0.9 作为 cutoff 参数值，则 cluster 函数将所有变量分为 10 类，如下所示。

```
T=cluster(Z, 0.9)
T =
     1
     1
     2
     3
     5
     6
     3
     1
     1
    10
     4
     3
     4
     3
     3
     4
     7
     6
```

```
      6
      8
      9
```

如果选择不连续系数 0.7 作为 cutoff 参数的值，则所有数据分为 16 类。

```
T=cluster(Z, 0.7)
T =
      1
      2
      4
      5
     10
     11
      6
      3
      1
     16
      8
      7
      9
      5
      6
      8
     13
     11
     12
     14
     15
```

② 通过指定类数进行聚类：如果将所有变量归为 3 类，将 cutoff 参数设置为 3，用 cluster 函数进行分类。

```
T=cluster(Z,3)
T =
      1
      1
      1
      1
      1
      1
      1
      1
      1
      3
      1
      1
      1
      1
      1
      1
```

```
1
1
1
1
2
```

9.3 K 均值聚类

K 均值聚类与系统聚类有所区别。进行 K 均值聚类时，函数 kmeans 将数据中的观测量分成 K 个相互排斥的类，并返回一个指数向量。与 linkage 函数所使用的系统聚类方法不同，kmeans 函数不用树结构描述数据中的组，而是创建单一水平的类。另一个不同在于，K 均值聚类使用实测值，而不是它们的近似值。

这些不同意味着 kmeans 函数对于大量数据的分类问题更合适。kmeans 函数认为数据中的每个观测量在空间中有一个位置。它通过寻找数据之间的一种关系，即每一类中的对象之间靠得尽可能地近，类与类之间的对象离得尽可能地远，从而能更容易地实现类的划分。可以从 5 种不同的距离度量中间进行选择，根据数据类型的不同采用不同的距离。在划分过程中，每一类由它的成员对象和它的质心或中心定义。每一类的质心是各对象到该点的距离之和最小的点。对于不同的距离度量方法，kmeans 计算质心的方法也不同。对于所有分类，kmeans 使用一种迭代算法，使每一个对象到它的质心的距离的和最小。该算法在各类之间移动对象，直到和不能再减小为止。在分类过程中，可以通过设置一些可选参数进行更多的控制，包括设定质心的初值和迭代的最大次数等。

本例通过比较将一套四维数据分为 3 类、4 类和 5 类的结果来获得可能的分类结果。

注意：因为本例中的每一个部分按顺序生成随机数，即，在设置新的随机数种子的情况下，必须按次序复制下面的所有步骤。如果不按步骤进行，虽然结果一致，但中间结果，如迭代次数或轮廓图的次序可能会不同。

第 1 步，装载数据。

```
load kmeansdata;
size(X)
ans =
   560    4
```

尽管这些数据是四维的，很难可视化，但通过 kmeans 函数可以探察到存在于它们中间的分组结构。将 kmeans 函数的 k 参数设为 3，指定将数据分为 3 类。对于本例，指定用城市街区的距离进行度量。使用随机选取的数据点的重心作为初始化的重心。

idx3 = kmeans(X,3,'distance','city');

要知道分类结果的合理程度，可以使用 kmeans 函数的分类指数输出绘制轮廓图。轮廓图用分类指数显示每个点的接近程度。轮廓图演示某类中的每个点与相邻类的中点的接近程度。该度量界于 1 和-1 之间。等于+1 时，表示点与相邻类中的点的距离很远；等于 0 时，表示该点的类属关系还不明确；等于-1 时，表示该点的分类可能是错误的。silhouette 函数在第 1 个输出参数中返回这些值。

```
[silh3,h] = silhouette(X,idx3,'city');
xlabel('Silhouette Value')
ylabel('Cluster')
```

生成的三分类轮廓图如图 9-3 所示。

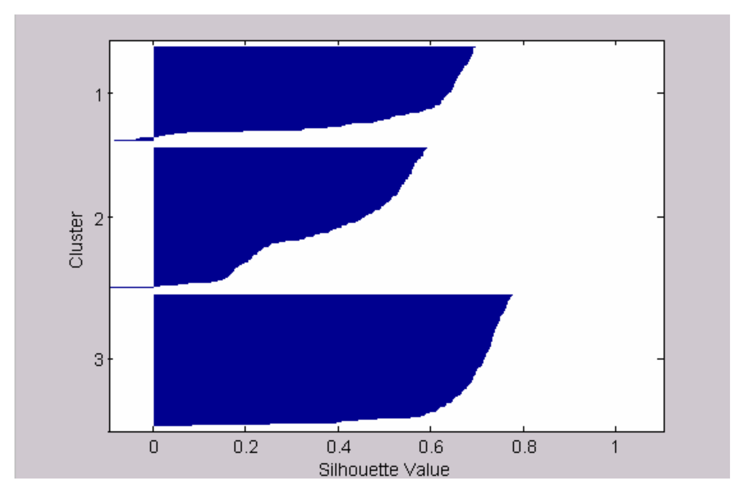


图 9-3　三分类的轮廓图

从轮廓图中可以看出，第 3 类中的大部分点的 silouette 值比较大（大于 0.6）表示该类在某种程度上区分于相邻的类。但是，第 2 类包含了许多具有较低 silhouette 值的点，而且第 1 类包含了少数几个具有负值的点，表示这两个类的区分不是很好。

第 2 步，确定正确的分类个数。

增加分类个数，看 kmeans 函数能否找到一个更好的数据分类。这次，使用可选的'display'参数来打印每次迭代的信息。

```
idx4 = kmeans(X,4, 'dist','city', 'display','iter');
  iter  phase   num      sum
    1     1     560    2897.56
    2     1      53   2736.67
    3     1      50   2476.78
    4     1     102    1779.68
    5     1       5     1771.1
    6     2       0     1771.1
6 iterations, total sum of distances = 1771.1
```

注意：kmeans 函数重新指定类间的点并重新计算类的重心时，距离的总和随迭代次数的增加逐渐减小。在本例中，算法设置在第 2 阶段没有发生任何改变，表示经过 5 次迭代以后，第 1 阶段已经达到最小。在有些问题中，第 1 阶段不会达到最小，但第 2 阶段总会达到。本解的轮廓图显示四分类比前面的三分类效果要好一些。

```
[silh4,h] = silhouette(X,idx4,'city');
xlabel('Silhouette Value')
ylabel('Cluster')
```

生成四分类的轮廓图，如图 9-4 所示。

一个更为量化的方法是比较两个分类的 silhouette 值的平均值。

```
mean(silh3)
ans =
     0.52594
mean(silh4)
```

```
ans =
     0.63997
```

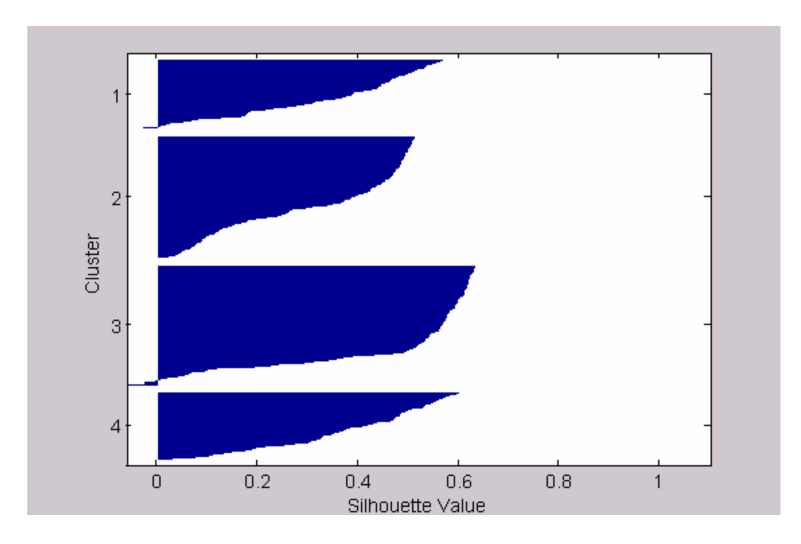


图 9-4　四分类轮廓图

最后，试图将数据分成 5 类。

```
idx5 = kmeans(X,5,'dist','city','replicates',5);
[silh5,h] = silhouette(X,idx5,'city');
xlabel('Silhouette Value')
ylabel('Cluster')
mean(silh5)
ans =
    0.52657
```

生成五分类的轮廓图，如图 9-5 所示。

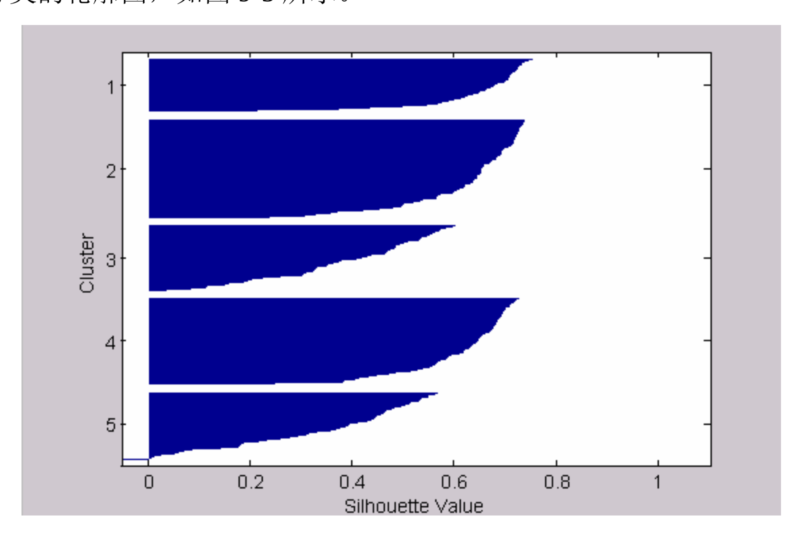


图 9-5　五分类的轮廓图

本轮廓图显示五分类可能不是好的选择，因为有两个分类中包含了 silhouette 值很低的点。在不知道数据到底要分成几类时，在一定范围内变换 k 值进行试验是一个好主意。象许多其

他类型的数值最小化一样，kmeans 函数的解经常取决于初值。使用 kmeans 函数时，有可能会获得一个局部解。为新类重新指定任何一个点时，都将增大点到重心距离的和。但这里存在一个更好的解，可以使用可选项'replicates'参数克服这个问题。对于四分类，指定 5 个 replicates，使用'display'参数为每个解打印出距离最终的和。

```
[idx4,cent4,sumdist] = kmeans(X,4,'dist','city',...
                              'display','final','replicates',5);
17 iterations, total sum of distances = 2303.36
 5 iterations, total sum of distances = 1771.1
 6 iterations, total sum of distances = 1771.1
 5 iterations, total sum of distances = 1771.1
 8 iterations, total sum of distances = 2303.36
```

输出表明，即使对于这个相对简单的问题，非全局最小化问题也存在。这 5 个 replicates 中的每一个都从一个不同的初始重心的不同随机选择集开始，并且 kmeans 函数发现两个不同的局部最小解。但是，kmeans 函数的最终结果是所有 replicates 中具有最小距离总和的那一个。

```
ans =
      1771.1
```

9.4 主成分分析

主成分分析是把多个指标化为少数几个综合指标的一种统计分析方法。在多变量的研究中，往往由于变量个数太多，并且彼此之间存在一定的相关性，使得所观测的数据在一定程度上反映的信息有所重叠。利用主成分分析则可以将这一问题化简，即通过降维，找到几个综合因子来代表原来众多的变量，使这些综合因子能尽可能地反映原来变量的信息量，而且彼此之间互不相关。

9.4.1 有关函数介绍

1．barttest 函数

利用该函数进行 Bartlett 维数检验，其调用格式如下：

- ndim = barttest(x, alpha) 用给定的显著性概率 alpha，返回维数，用于解释 x 数据矩阵的非随机变化特征。维数由一系列假设检验确定。假设 ndim=1 的检验是检验与每个因子一起的方差是否相等；ndim=2 的检验则检验第 2 个因子至最后一个因子一起的方差是否相等，依次类推。
- [ndim, prob, chisquare] = barttest(x, alpha) 返回维数，假设检验的显著性概率和与检验相关的卡方值。

【例 9-12】

```
x = mvnrnd([0 0], [1 0.99; 0.99 1],20);
x(:,3:4) = mvnrnd([0 0], [1 0.99; 0.99 1],20);
x(:,5:6) = mvnrnd([0 0], [1 0.99; 0.99 1],20);
[ndim, prob] = barttest(x,0.05)
ndim =
   3
prob =
```

```
         0
         0
         0
    0.5081
    0.6618
    1.0000
```

2．pcacov 函数

利用该函数，使用协方差矩阵进行主成分分析，其调用格式如下：

● [pc, latent, explained] = pcacov(X) 利用协方差矩阵 X，返回主要因子 pc、协方差矩阵 X 的特征值 latent、观测量中由每一个特征向量所解释的总方差的百分比 explained。

【例 9-13】

```
load hald
covx = cov(ingredients);
[pc,variances,explained] = pcacov(covx)
pc =
    0.0678   -0.6460    0.5673   -0.5062
    0.6785   -0.0200   -0.5440   -0.4933
   -0.0290    0.7553    0.4036   -0.5156
   -0.7309   -0.1085   -0.4684   -0.4844
variances =
  517.7969
   67.4964
   12.4054
    0.2372
explained =
   86.5974
   11.2882
    2.0747
    0.0397
```

3．pcares 函数

利用该函数计算源于主成分分析的残差，其调用格式如下：

● pcares(X, ndim) 通过保留 X 的 ndim 个因子成分来获得残差。注意，ndim 为标量并且必须小于 X 的列数。将数据矩阵、协方差矩阵与该函数一起使用。

【例 9-14】 当因子维数由 1 个增加为 3 个时，显示 Hald 数据第 1 行的残差的下降情况。

```
load hald
r1 = pcares(ingredients,1);
r2 = pcares(ingredients,2);
r3 = pcares(ingredients,3);
r11 = r1(1,:)
r11 =
    2.0350    2.8304   -6.8378    3.0879
r21 = r2(1,:)
r21 =
   -2.4037    2.6930   -1.6482    2.3425
```

```
r31 = r3(1,:)
r31 =
    0.2008    0.1957    0.2045    0.1921
```

4．princomp 函数

利用该函数进行主成分分析，其调用格式如下：

● [PC, SCORE, latent, tsquare] = princomp(X) 根据数据矩阵 X 返回因子成分 PC、Z 分数 SCORE、X 的协方差矩阵的特征值 latent 和 Hotelling 的 T^2 统计量 tsquare。Z 分数是通过将原始数据转换到因子成分空间中得到的数据。latent 向量的值为 SCORE 的列数据的方差。Hotelling 的 T^2 为来自数据集合中心的每一个观测量的多变量距离的度量。

【例 9-15】 为 Hald 数据集合中的 ingredients 数据计算因子成分，及由每个成分解释的方差。

```
load hald;
[pc,score,latent,tsquare] = princomp(ingredients);
pc,latent
pc =
    0.0678   -0.6460    0.5673   -0.5062
    0.6785   -0.0200   -0.5440   -0.4933
   -0.0290    0.7553    0.4036   -0.5156
   -0.7309   -0.1085   -0.4684   -0.4844
latent =
  517.7969
   67.4964
   12.4054
    0.2372
```

9.4.2 应用综合实例

【例 9-16】 应用 MATLAB 内部的数据 cities.mat 进行分析。该数据是美国 329 个城市反映生活质量的 9 项指标的数据。这 9 项指标分别为：气候、住房、健康状况、犯罪、交通、教育、艺术、娱乐和经济。

首先装载数据文件

```
load cities
whos
    Name            Size          Bytes  Class
  categories        9x14            252  char array
  names           329x43          28294  char array
  ratings         329x9           23688  double array
Grand total is 17234 elements using 52234 bytes
```

分别为类别、名称和比率的大小和数组类型和大小。

```
categories
categories =
climate
housing
health
crime
```

```
transportation
education
arts
recreation
economics
```

可见类别（categories）中包含评价生活质量的 9 项指标名称。

现在看看 names 变量的前面几行：

```
first5=names(1:5,:)
first5 =
  Abilene, TX
  Akron, OH
  Albany, GA
  Albany-Troy, NY
  Albuquerque, NM
```

用箱形图来表达

```
boxplot(ratings,0,'+',0)
set(gca,'YTicklabel',categories)
```

从图 9-6 中可以看出，艺术和住房的变化最大，气候的变化最小。

当原始数据的量级和量纲存在较大差异时，需要先对数据进行标准化，然后进行主成分分析。标准化的方法是将原始数据的各列除以各列的标准差。

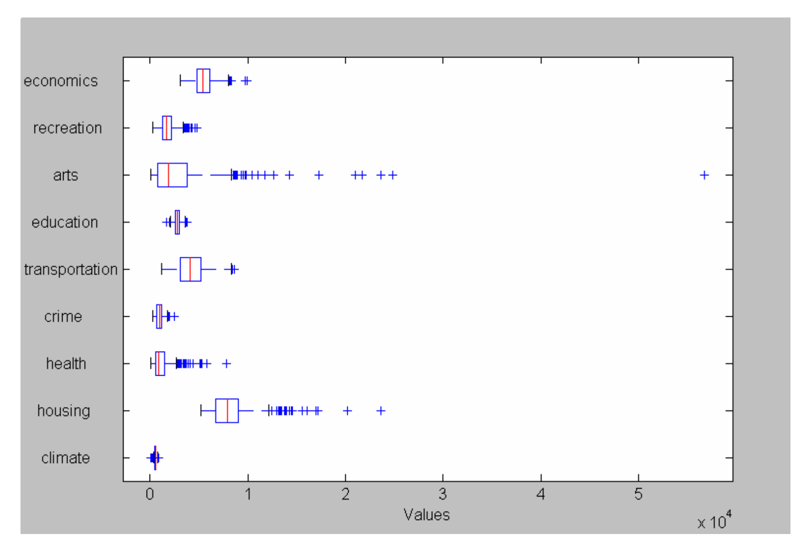


图 9-6　各指标的箱形图

```
stdr=std(ratings);
sr=ratings./stdr(ones(329,1),:);
```

现在寻找主成分

```
[pcs,newdata,variances,t2]=princomp(sr);
```

（1）第 1 个输出——主成分（pcs）

pcs 包括 9 个主成分。下面列出前 3 个。

```
p3=pcs(:,1:3)
p3 =
```

```
    0.2064    0.2178   -0.6900
    0.3565    0.2506   -0.2082
    0.4602   -0.2995   -0.0073
    0.2813    0.3553    0.1851
    0.3512   -0.1796    0.1464
    0.2753   -0.4834    0.2297
    0.4631   -0.1948   -0.0265
    0.3279    0.3845   -0.0509
    0.1354    0.4713    0.6073
```

可以看出，第 1 个主成分中第 7 个元素的权重最大。

可以通过查看 p3 乘以 p3 转置的结果来判断其正交性。

```
I=p3'*p3
I =
    1.0000   -0.0000   -0.0000
   -0.0000    1.0000    0.0000
   -0.0000    0.0000    1.0000
```

计算结果为单位矩阵，说明各主成分之间满足正交性。

（2）第 2 个输出——主成分得分（newdata）

主成分得分是原始数据在由主成分所定义的新坐标系中的确定的数据，其大小与输入数据矩阵的大小相同。

下面的代码生成的图形（图 9-7）显示了 newdata 的前两列数据作为前两个主成分时的结果。

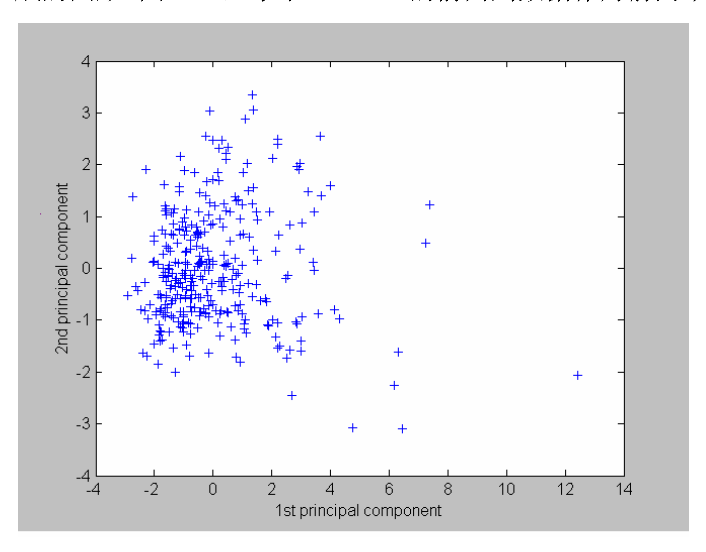


图 9-7　数据在前两个主成分构成的体系内的分布

```
plot(newdata(:,1),newdata(:,2),'+')
xlabel('1st principal component');
ylabel('2nd principal component');
```

图中右侧显示的是异常值。可以利用 gname 函数标注图中的点。下面用字符串矩阵 names 调用 gname 函数，该矩阵中包含图中所有点的案例标签。

gname(names)

运行该命令，将在图中生成一个十字形交叉线，交叉点跟随鼠标移动。在散点附近单击，

将标注该点的字符串标签。标注结束以后，单击回车键。图 9-8 显示了标注部分异常值以后的结果。

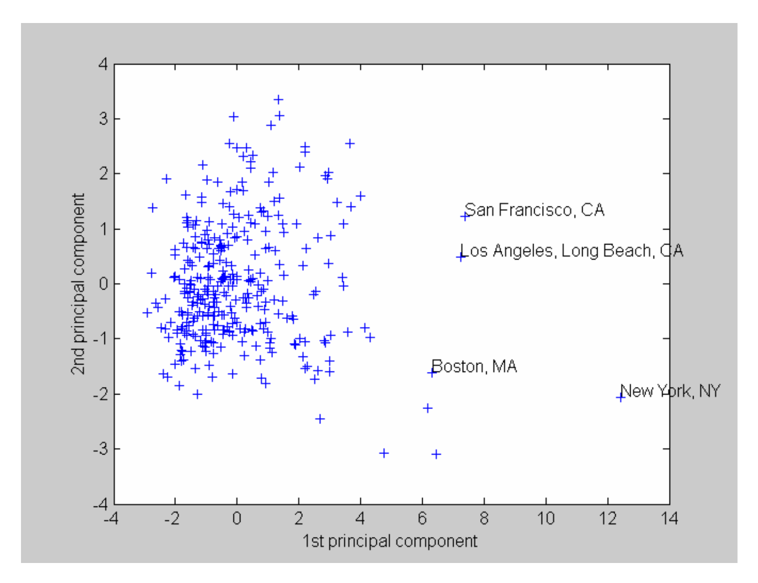


图 9-8　用城市名称标注图中的点

若调用 gname 函数时不带变量，则系统用案例行号来作为标签。

```
gname
```

重复以上操作，生成如图 9-9 所示的结果。各标签为对应案例的行号。

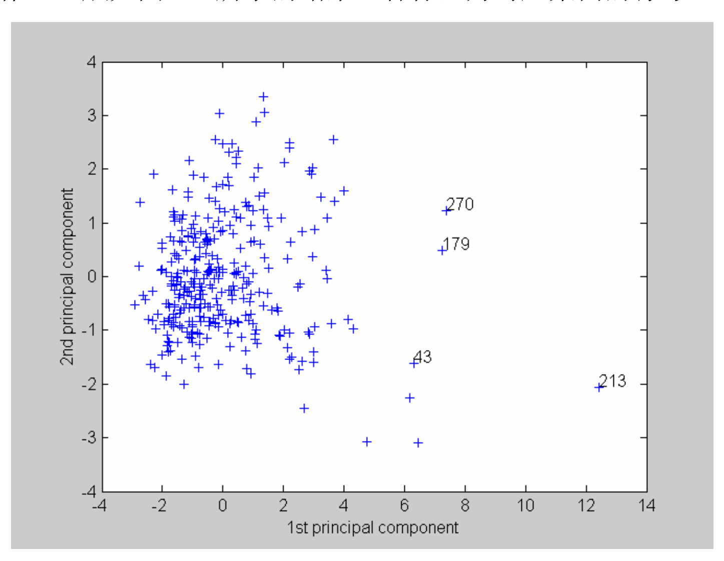


图 9-9　用案例号标注图中的点

可以创建一个包含所有选择行号的索引变量，在图中进行标注：

```
metro=[43 65 179 213 234 270 314];
names(metro,:)
ans =
  Boston, MA
  Chicago, IL
```

```
Los Angeles, Long Beach, CA
New York, NY
Philadelphia, PA-NJ
San Francisco, CA
Washington, DC-MD-VA
```

从 ratings 矩阵中删除这些行：

```
rsubset=ratings;
nsubset=names;
nsubset(metro,:)=[];
rsubset(metro,:)=[];
size(rsubset)
ans =
      322    9
```

（3）第 3 个输出——主成分方差（variances）

主成分方差 variances 是由 newdata 的对应列所解释的包含方差的向量。

```
variances
variances =
    3.4083
    1.2140
    1.1415
    0.9209
    0.7533
    0.6306
    0.4930
    0.3180
    0.1204
```

可以很方便地计算由每个主成分所解释的总方差的百分数。

```
percent_explained=100*variances/sum(variances)
percent_explained =
   37.8699
   13.4886
   12.6831
   10.2324
    8.3698
    7.0062
    5.4783
    3.5338
    1.3378
```

可见，前面 5 个主成分所解释的方差占了总方差的 80%以上。

用帕累托图可以描述每个主成分所占的百分数。

```
pareto(percent_explained)
xlabel('Principal Component')
ylabel('Variance Explained(%)')
```

生成图 9-10。从图中可以看出，前面 3 个主成分基本上解释了 2/3 的标准化 ratings 的总变异性。

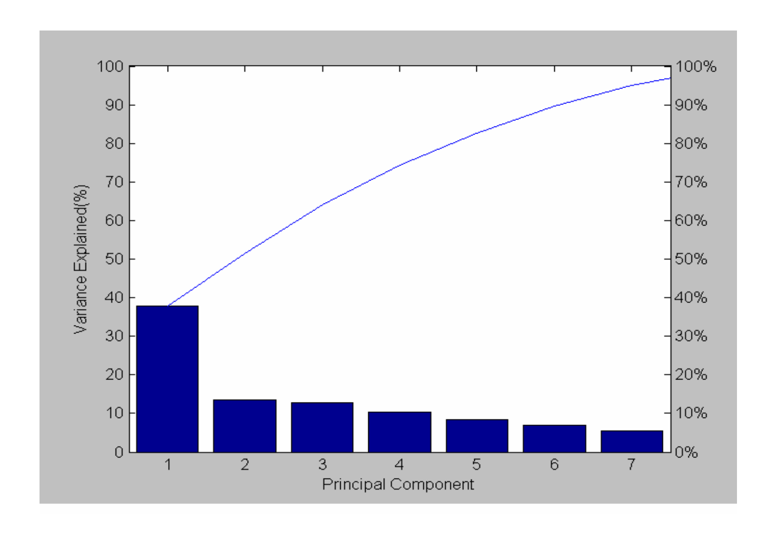


图 9-10　各主成分解释方差的帕累托图

（4）第 4 个输出—Hoteling 的 T^2 检验（t2）

Hoteling 的 T^2 检验统计量是描述每一测量值与数据中心距离的统计量，用它可以找到数据中的极值点。

```
[st2,index]=sort(t2);        %升序排序
st2=flipud(st2);             %降序排序的数据值
index=flipud(index);         %降序排列的系数
extreme=index(1)
extreme =
   213
names(extreme,:)
ans =
New York, NY
```

说明 New York 是距离数据中心最远的城市。

【例 9-17】　为了进行土壤分析，研究土壤质量，抽取了 20 个样本，每个样本有 4 个指标：淤泥含量、粘土含量、有机物、酸性指标 pH 值。原始数据见表 9-4。

表 9-4　原始数据表

编　号	淤泥含量(%)	粘土含量(%)	有机物(%)	PH 值
1	13.0	9.7	1.5	6.4
2	10.0	7.5	1.5	6.5
3	20.6	12.5	2.3	7.0
4	33.3	19.0	2.8	5.8
5	20.5	14.2	1.9	6.9
6	10.0	6.7	2.2	7.0
7	12.7	5.7	2.9	6.7
8	36.5	15.7	2.3	7.2
9	37.1	14.3	2.1	7.2
10	25.5	12.9	1.9	7.3
11	26.5	14.9	2.4	6.7

续表

编　号	淤泥含量(%)	粘土含量(%)	有机物(%)	PH 值
12	22.3	8.4	4.0	7.0
13	30.8	7.4	2.7	6.4
14	25.3	7.0	4.8	7.3
15	31.2	11.6	2.4	6.3
16	22.7	10.1	3.3	6.2
17	31.2	9.6	2.4	6.0
18	13.2	6.6	2.0	5.8
19	11.1	6.7	2.2	7.2
20	20.7	9.6	3.1	5.9

进行主成分分析之前，如果各变量的数量级和量纲等存在较大的差异，则需要首先进行数据标准化。本例中数据差异较小，不必进行标准化。

```
X=[13.0     9.7     1.5     6.4
    1.0     7.5     1.5     6.5
   20.6    12.5     2.3     7.0
   33.3    19.0     2.8     5.8
   20.5    14.2     1.9     6.9
   10.0     6.7     2.2     7.0
   12.7     5.7     2.9     6.7
   36.5    15.7     2.3     7.2
   37.1    14.3     2.1     7.2
   25.5    12.9     1.9     7.3
   26.5    14.9     2.4     6.7
   22.3     8.4     4.0     7.0
   30.8     7.4     2.7     6.4
   25.3     7.0     4.8     7.3
   31.2    11.6     2.4     6.3
   22.7    10.1     3.3     6.2
   31.2     9.6     2.4     6.0
   13.2     6.6     2.0     5.8
   11.1     6.7     2.2     7.2
   20.7     9.6     3.1     5.9];
[pcs,newdata,variances,t2]=princomp(X)
pcs =
   0.9656   -0.2536   -0.0560    0.0072
   0.2592    0.9552    0.1413   -0.0203
   0.0177   -0.1524    0.9756   -0.1570
   0.0011   -0.0027    0.1585    0.9874
```

所以，新组成的 4 个主成分 PC1，PC2，PC3 和 PC4 可以写成（假设淤泥含量为 X1、粘土含量为 X2、有机物为 X3、PH 值为 X4）：

```
PC1=0.9656*X1+0.2592*X2+0.0177*X3+0.0011*X4
PC2=-0.2536*X1+0.9552*X2-0.1524*X3-0.0027*X4
PC3=-0.0560*X1+0.1413*X2+0.9756*X3+0.1585*X4
PC4=0.0072*X1-0.0203*X2-0.1570*X3+0.9874*X4
```

可见第 1 主成分中第 1 个元素（对应于淤泥含量）的权重最大。

```
newdata =
      -9.1692       1.7382      -0.6429      -0.1250
     -21.3272       2.6802      -0.2657      -0.0682
      -1.0896       2.3615       0.2026       0.3398
      12.8668       5.2762       0.7072      -0.9638
      -0.7526       4.0720       0.0423       0.2687
     -12.8308      -0.4748      -0.1207       0.3968
     -10.4707      -2.2208       0.2222       0.0305
      15.0941       1.3847      -0.2042       0.5871
      15.3070      -0.0742      -0.6307       0.6513
       3.7391       1.5609      -0.3580       0.7261
       5.2314       3.1431       0.2612       0.0218
      -0.4807      -2.2452       1.1867       0.1685
       7.4443      -5.1565      -0.7940      -0.1382
       2.0678      -4.4662       1.6489       0.3892
       8.9140      -1.2001      -0.5316      -0.2722
       0.3330      -0.6139       0.5948      -0.5431
       8.3952      -3.1097      -0.8617      -0.5278
      -9.7715      -1.3482      -0.6993      -0.7315
     -11.7684      -0.7544      -0.1506       0.6022
      -1.7318      -0.5529       0.3935      -0.8122
variances =
  101.6280
    7.6502
    0.4489
    0.2629
```

为 newdata 中各列数据对应的方差。

```
t2 =
    2.2022
    5.5896
    1.2713
    9.9150
    2.4515
    2.2807
    1.8370
    3.8963
    4.8054
    2.7467
    1.7145
    3.9059
    5.4980
    9.2816
    1.8814
    1.9603
    4.6709
    4.3013
```

```
2.8670
2.9234
```

t2 显示第 4 个和第 14 个案例数据距离数据中心的距离最远。

```
percent_explained=100*variances/sum(variances);
pareto(percent_explained)
xlabel('Principal Component')
ylabel('Variance Explained(%)')
```

从图 9-11 中可以看出，由第 1 个主成分所解释的方差占到总方差的 90%以上。第 3 主成分和第 4 主成分所解释的方差可以忽略不计。

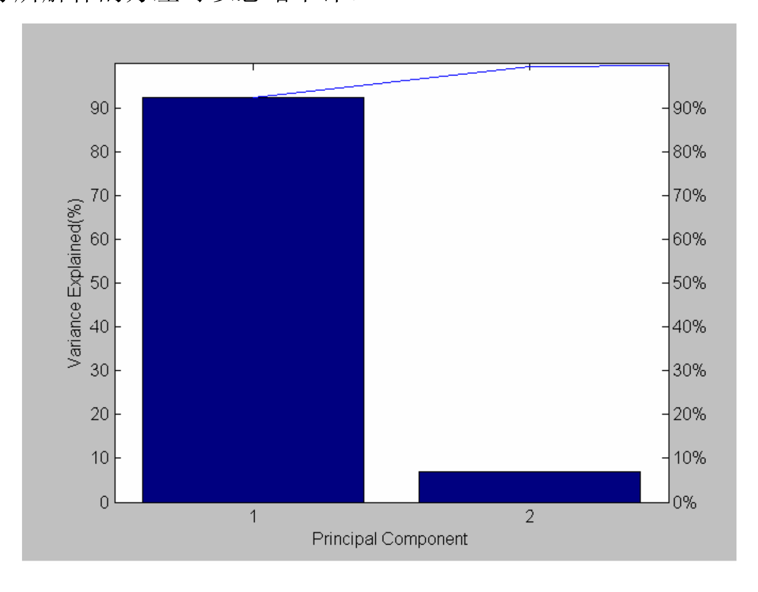


图 9-11　主成分解释方差的帕累托图

9.5　因子分析

因子分析是拟合多变量数据模型的一种途径，它衡量独立变量的种类。在因子分析模型中，度量变量取决于个数更少的非测量因子，因为每个因子都会共同影响部分变量，它们被称为公共因子。假设每个变量是公共因子的线性组合，系数称为荷载。每个度量变量也包括一个独立随机变异性解释的组分，称为“特定方差”，因为它对于某个变量而言是特定的。

【例 9-18】　找到影响股票价格的公共因子

在 100 周时间内，记录了 10 个公司股票价格的百分比。在这 10 个公司中，前 4 个可被归类为技术型，另 3 个为金融型，剩下的 3 个为零售型。在同一部门，公司股票价格随经济情况的改变而共同改变是合理的。因子分析可以提供各公司在某部门内每周股票价格作相似变化的证据。

首先装载数据，然后调用 factoran 函数，并指定一个 3 个公共因子的模型拟合。默认时，factoran 函数将进行旋转以后荷载的大小估计，以试图使它们的解释相近。但此时，指定一个未旋转的解。

```
load stockreturns
[Loadings,specificVar,T,stats] = factoran(stocks,3,'rotate','none');
```

factoran 函数返回的前两个变量是估计的荷载和估计的指定方差。荷载矩阵的每一行代表 10 支股票中的 1 支，每一列对应于一个公共因子。对于没有旋转的估计，拟合中因子的解释比较困难，因为大多数股票有两个或两个以上的因子包含相当大的系数。

```
Loadings
Loadings =
    0.8885      0.2367      -0.2354
    0.7126      0.3862       0.0034
    0.3351      0.2784      -0.0211
    0.3088      0.1113      -0.1905
    0.6277     -0.6643       0.1478
    0.4726     -0.6383       0.0133
    0.1133     -0.5416       0.0322
    0.6403      0.1669       0.4960
    0.2363      0.5293       0.5770
    0.1105      0.1680       0.5524
```

注意：因子旋转帮助简化荷载矩阵的结构，所以，给因子指定一个有意义的解释将更容易。从估计的特定方差，我们可以看到，有一支股票的价格变化相当大。

```
specificVar
specificVar =
     0.0991
     0.3431
     0.8097
     0.8559
     0.1429
     0.3691
     0.6928
     0.3162
     0.3311
     0.6544
```

特定方差为 1 时表示在该变量中没有公共因子组分，特定方差为 0 时则表示变量完全由公共因子确定。这些数据看起来落在它们之间的某个地方。stats 结构中返回的 p 值显示不能拒绝 3 个公共因子的零假设。零假设为本模型对这些数据的变异性提供了一个满意的解释。

```
stats.p
ans =
     0.8144
```

要确定因子少于 3 个时是否能提供一个可以接受的拟合，可以用两个公共因子建立一个模型。第 2 次拟合的 p 值高度显著，拒绝两个因子的零假设，表示相似的模型不足以解释这些数据的模式。

```
[Loadings2,specificVar2,T2,stats2] = factoran(stocks, 2,...
                                   'rotate','none');
stats2.p
ans =
    3.5610e-006
```

就像上面的结果所演示的，原来未旋转的因子分析拟合的荷载估计可以具有一个复杂的结构。因子旋转的目的就是要找到一个参数化的效果。其中，每个变量仅有少量的大荷载，

即，只被少量因子影响，最好只有一个。这样有利于解释因子所表示的含义。如果你认为荷载矩阵的每一行是 M 维空间中的一个点，则每个因子对应于一个坐标轴。因子旋转等价于旋转这些轴，并在旋转的坐标系统中计算这些新的荷载。有些方法让坐标轴正交，其他一些倾斜的方法则改变它们之间的角度。对于本例，通过使用 promax 准则（一个公共的倾斜法）来旋转估计的荷载。

```
[LoadingsPM,specVarPM] = factoran(stocks,3,'rotate','promax');
LoadingsPM
LoadingsPM =
    0.9452      0.1214     -0.0617
    0.7064     -0.0178      0.2058
    0.3885     -0.0994      0.0975
    0.4162     -0.0148     -0.1298
    0.1021      0.9019      0.0768
    0.0873      0.7709     -0.0821
   -0.1616      0.5320     -0.0888
    0.2169      0.2844      0.6635
    0.0016     -0.1881      0.7849
   -0.2289      0.0636      0.6475
```

promax 旋转创建了一个简单的荷载结构，对于其中的一个，大部分股票只在一个因子上具有大的荷载。为更清楚地看到这一点，可以将因子荷载作为坐标绘出每支股票的图。本图中的荷载数据简单结构显示股票落在某个因子对应的坐标轴附近。因为有 3 个因子，用两两之间的二维图更能展示荷载。

```
subplot(1,2,1);
plot(LoadingsPM(:,1),LoadingsPM(:,2),'b.');
text(LoadingsPM(:,1),LoadingsPM(:,2), num2str((1:10)'));
line([-1 1 NaN 0 0 NaN 0 0],[0 0 NaN -1 1 NaN 0 0],...
    'Color','black');
xlabel('Factor 1');
ylabel('Factor 2');
axis square;

subplot(1,2,2);
plot(LoadingsPM(:,1),LoadingsPM(:,3),'b.');
text(LoadingsPM(:,1),LoadingsPM(:,3), num2str((1:10)'));
line([-1 1 NaN 0 0 NaN 0 0],[0 0 NaN -1 1 NaN 0 0],...
    'Color', 'black');
xlabel('Factor 1');
ylabel('Factor 3');
axis square;
```

生成的图形如图 9-12 所示。

图 9-12 显示 promax 已经成功地把因子荷载旋转到更简单的结构。每支股票主要依赖于一个因子，所以用它影响的股票描述每个因子成为可能。基于哪个公司接近哪个轴的情况，可以得出第 1 个因子表示金融部门，第 2 个表示零售部门，第 3 个为技术部门。原先关于股票价格主要在部门中间变化的推测显然被当前数据所支持。

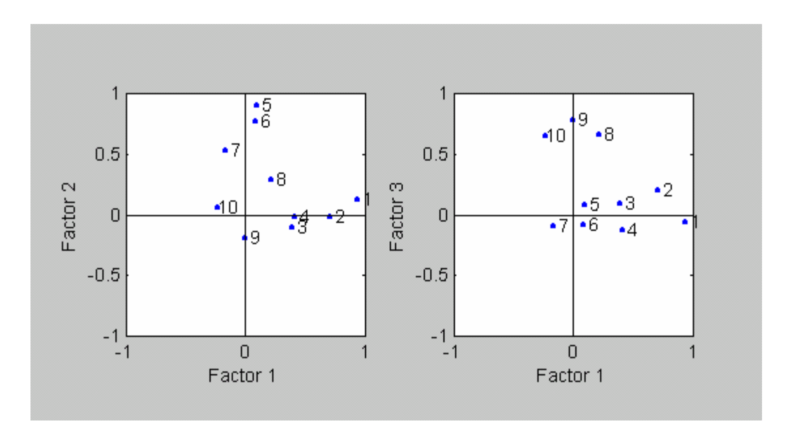


图 9-12　转换后的荷载分布

有时，在测量数据因子得分的基础上进行分类是很有用的。例如，如果你接受的因子模型和旋转因子的解释，你可能希望用 3 种股票对应的部门中的一个如何受欢迎来对每一周的情况进行分类。因为本例中的数据是原始的股票价格变化数据，而不是它们的相关矩阵，可以用 factoran 函数返回对每一周 3 个旋转公共因子中每一个的值的估计。然后，可以绘制得分图，看每一周不同的部门是如何受到影响的。

```
[LoadingsPM,specVarPM,TPM,stats,F] = factoran(stocks, 3,...
                                        'rotate','promax');
subplot(1,1,1);
plot3(F(:,1),F(:,2),F(:,3),'b.');
line([-4 4 NaN 0 0 NaN 0 0], [0 0 NaN -4 4 NaN 0 0],...
    [0 0 NaN 0 0 NaN -4 4], 'Color','black');
xlabel('Financial Sector');
ylabel('Retail Sector');
zlabel('Technology Sector');
grid on;
axis square;
view(-22.5, 8);
```

生成的得分图如图 9-13 所示。

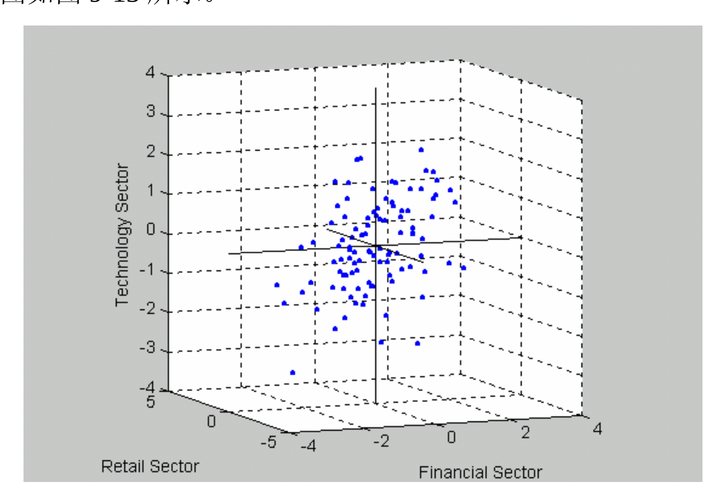


图 9-13　得分图

倾斜旋转经常创建相关因子。图 9-13 显示了第 1 个和第 3 个因子之间相关的一些证据，可以通过计算估计的因子相关矩阵来做更多的探察。

```
inv(TPM'*TPM)
ans =
      1.0000    0.1559    0.4082
      0.1559    1.0000   -0.0559
      0.4082   -0.0559    1.0000
```

9.6 多元方差分析

9.6.1 单因素多元方差分析

单因素多元方差分析检验某变量是否受到其他一个或多个变量的影响。利用该分析过程，可以分析因素之间的主效应，也可以分析因素之间的交互效应。

用 manova1 函数进行单因素多元方差分析(MANOVA)，其调用格式如下：

● d = manova1(X, group) 为了比较由 group 参数分组的 X 各列的多变量均值，进行单因素多元方差分析。X 是数据值的 $m\times n$ 矩阵，对于无重复的测量，每一行是一个基于 n 个变量的观测向量。group 为一分组变量，定义为向量、字符串数组和字符串单元数组。如果在 group 数组中有相同的值，两次观测的结果将位于同一组中。每一组中的观测值代表一个取自总体的样本。该函数返回 d，它是包含组均值的空间维数的估计。

manova1 函数检验每个组的均值为相同 n 维多元向量，且从样本 $\boldsymbol{X}$ 中得到的任何差异归结于随机因素的零假设是否成立。如果 $d = 0$，则在 5%的水平上接受零假设；若 $d = 1$，则在 5%的水平上拒绝零假设，但是不能拒绝多变量均值位于相同直线上的假设。近似地有，如果 $d = 2$，则在 n 维空间中，多变量均值可能位于相同的平面上，但没有位于同一条直线上。

● d = manova1(X, group, alpha) 给出显著性水平的控制参数 alpha。返回值 d 将是 p > alpha 时最小的维数，其中 p 为检验均值是否位于该维数的空间上的 p 值。

● [d, p] = manova1(...) 同样返回一个 p 参数，它是一个包含均值是否位于 0 维、1 维、2 维等空间的 p 值的向量，最大可能维数是空间维数或组数减 1。对于每一维，有一个 p 值达到但不包含最大值。

如果第 i 个 p 值接近于 0，则认为组均值落在第 i-1 空间的零假设可疑。p 值的大小决定结果是否“统计上显著”，其具体取值由用户通过输入变量 alpha 给定。通常认为 p 值小于 0.05 或 0.01 时，结果是显著的。

● [d, p, stats] =manova1(...) 还返回 stats 参数，它是一个包含其他多元方差分析结果的结构。该结构包含表 9-5 中的一些信息。

表 9-5 stats 结构的选项

域　　值	内　　容
W	组内平方和和交叉矩阵
B	组间平方和和交叉矩阵

续表

域　　值	内　　容
T	总的平方和和交叉矩阵
dfW	W 的自由度
dfB	B 的自由度
dfT	T 的自由度
lambda	Wilk 的 λ 检验统计量值向量，该统计量检验均值是否具有维数 0，1 等
chisq	λ 至近似卡方分布的转换
chisqdf	chisq 的自由度
eigenval	$\boldsymbol{W}^{-1}\boldsymbol{B}$的特征值
eigenvec	$\boldsymbol{W}^{-1}\boldsymbol{B}$的特征值；为变量 *C* 的系数，它们进行了标准化，使得变量 *C* 的组内差为 1
canon	变量 ***C***，等于 XC*eigenvec，其中 XC 为 *X* 各列减去列均值以后得到的矩阵
mdist	每个点到其组均值的马氏距离向量
gmdist	每个组均值之间的马氏距离矩阵

变量 ***C*** 是原始变量的线性组合，可作为组间的最大间隔。特别地，C(:, 1)为 *X* 各列的线性组合，组间具有最大间隔。这意味着在所有可能的线性组合中，它是单因素方差分析中 *F* 统计量最显著的一个。C(:, 2)具有最大间隔，它与 C(:, 1)正交。

将单因素多元方差分析的输出与其他函数一起使用，可以帮助进行分析。例如，如果想用 gplotmatrix 函数将原始变量绘成分组散点图矩阵，可以将前两个 canonical 变量用 gscatter 函数对组间隔进行图形显示。可以用 manovacluster 函数绘树形图，显示组均值的分类。

注意：manoval 函数要求 ***X*** 中的数据符合下面的假设：

每个组的总体呈正态分布；

对于每个总体，其方差-协方差矩阵是相同的；

所有的观测量都是相互独立的。

【例 9-19】 下面用单因素多元方差分析来确定 4 种小汽车属性均值之间是否有差异，用小汽车的产地来进行数据分组。

```
load carbig
[d,p] = manova1([MPG Acceleration Weight Displacement],Origin)
d =
    3
p =
         0
    0.0000
    0.0075
    0.1934
```

输入矩阵有四维，所以组均值必须位于四维空间上。manova1 函数显示不能拒绝均值落在三维子空间的零假设。

【例 9-20】 健康人（类别 1）、硬化症患者（类别 2）和冠心病患者（类别 3）心电图的 5 项不同指标（X1～X5）如下所示，试进行多元方差分析。

类别	X1	X2	X3	X4	X5
1	8.11	261.0	13.23	5.46	7.36
1	9.36	185.39	9.02	5.66	5.99
1	9.85	249.58	15.61	6.06	6.11
1	2.55	137.13	9.21	6.11	4.35
1	6.01	231.34	14.27	5.21	8.79
1	9.64	231.38	13.03	4.88	8.53
1	4.11	260.25	14.72	5.36	10.02
1	8.90	259.51	14.16	4.91	9.79
1	7.71	273.84	16.01	5.15	8.79
1	7.51	303.59	19.14	5.70	8.53
1	8.06	231.03	14.41	5.72	6.15
2	6.80	308.90	15.11	5.52	8.49
2	8.68	258.69	14.02	4.97	7.16
2	5.67	355.54	15.13	4.97	9.43
2	8.10	476.69	7.38	5.32	11.32
2	3.71	316.12	17.12	6.04	8.17
2	5.37	274.57	16.57	4.98	9.67
2	9.89	409.42	19.47	5.19	10.49
3	5.22	330.34	18.19	4.96	9.61
3	4.71	331.47	21.26	4.30	13.72
3	4.71	352.50	20.79	5.07	11.00
3	3.36	347.31	17.90	4.65	11.19
3	8.27	189.59	12.74	5.46	6.94

在命令窗口输入下面的命令行：

```
X=[8.11     261.0     13.23     5.46     7.36
   6.80     308.90    15.11     5.52     8.49
   9.36     185.39     9.02     5.66     5.99
   8.68     258.69    14.02     4.97     7.16
   9.85     249.58    15.61     6.06     6.11
   5.67     355.54    15.13     4.97     9.43
   2.55     137.13     9.21     6.11     4.35
   8.10     476.69     7.38     5.32    11.32
   6.01     231.34    14.27     5.21     8.79

   3.71     316.12    17.12     6.04     8.17
   9.64     231.38    13.03     4.88     8.53
   5.37     274.57    16.57     4.98     9.67
   4.11     260.25    14.72     5.36    10.02
   9.89     409.42    19.47     5.19    10.49
   8.90     259.51    14.16     4.91     9.79
   5.22     330.34    18.19     4.96     9.61
   7.71     273.84    16.01     5.15     8.79
   4.71     331.47    21.26     4.30    13.72
   7.51     303.59    19.14     5.70     8.53
   4.71     352.50    20.79     5.07    11.00
   8.06     231.03    14.41     5.72     6.15
   3.36     347.31    17.90     4.65    11.19
   8.27     189.59    12.74     5.46     6.94];
group=[1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 3 1 3 1 3 1 3 3]';
```

```
[d,p,stats] = manova1(X,group)
d =
     1
p =
    0.0187
    0.1186
stats =
           W: [5x5 double]
           B: [5x5 double]
           T: [5x5 double]
         dfW: 20
         dfB: 2
         dfT: 22
      lambda: [2x1 double]
       chisq: [2x1 double]
     chisqdf: [2x1 double]
    eigenval: [5x1 double]
    eigenvec: [5x5 double]
       canon: [23x5 double]
       mdist: [23x1 double]
      gmdist: [3x3 double]
      gnames: {3x1 cell}
```

d=1，说明可以在 5%的水平上拒绝所有分组均值相等的零假设，但是不能拒绝各均值位于相同直线上的假设。第 1 个 *p* 值小于 0.05，所以 *d*=1。stats 向量中为分析统计量，其意义参见表 9-5。

9.6.2　分组聚类

可用 manovacluster 函数进行分组聚类。该函数进行多元方差分析以后，绘冰柱图，显示组均值的分类。其调用格式如下：

- manovacluster(stats) 进行多元方差分析以后，生成组均值的树形图。stats 为单因素多元方差分析的输出。通过对组均值之间的马氏距离矩阵用单向联结方法进行计算来分类。当组的个数很大时，使用 dendrogram 函数效果较好。
- manovacluster(stats, 'method') 使用指定的联结方法进行分类。'method'可以是表 9-6 所示的字符串中的一个。

表 9-6　距离方法选项表

字　符　串	含　　义
'single'	最短距离法(默认时选用)
'complete'	最长距离法
'average'	平均距离法
'centroid'	重心法
'ward'	平方和递增法

- H = manovacluster(stats, 'method') 返回冰柱图中直线的句柄向量。

【例 9-21】　下面利用一个更大的汽车数据集进行分析，确定哪些国家生产的汽车具有最

相近的属性。

```
load carbig
X = [MPG Acceleration Weight Displacement];
[d,p,stats] = manova1(X,Origin);
manovacluster(stats)
```

从图 9-14 所示的冰柱图中可以看出，日本(Japan)和德国(Germany)生产的汽车具有最相似的属性。

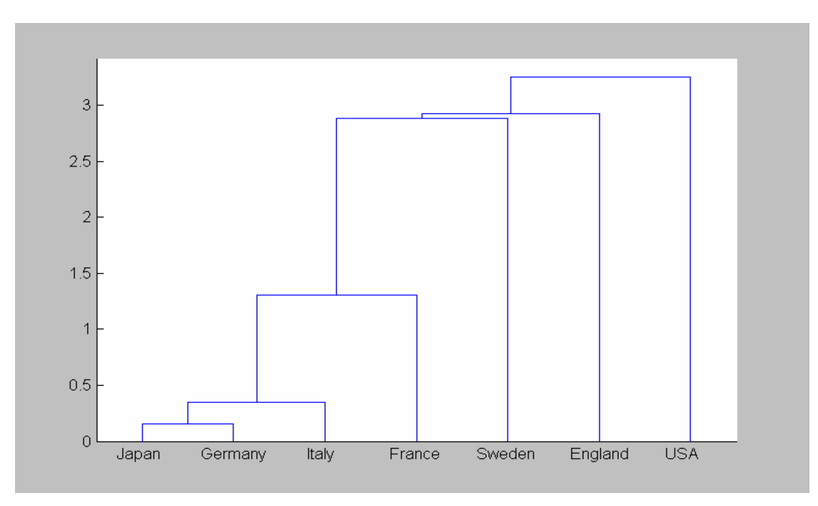


图 9-14　冰柱图

9.6.3　多元比较

在单因素方差分析中，比较多组数据的均值，检验它们相等的零假设是否成立。有时需要知道哪些成对均值具有显著差异，哪些却没有。可以提供此类信息的检验称为“多元比较过程”。用 multcompare 函数进行多元比较。

1．multcompare 函数

用 multcompare 函数进行均值或其他估计的多元比较检验，其调用格式如下：

● c = multcompare(stats) 用 stats 结构中的信息进行多元比较检验，返回成对比较结果的矩阵。它还显示一个代表检验结果的交互图形。

进行某个组均值与其他组均值之间的简单 t 检验时，指定一个显著性水平，确定 t 统计量的临界值。例如，你可以指定 alpha = 0.05，以保证没有差异时，得出差异显著的机会小于或等于 5%。当有多个组均值时，还会有许多成对数据需要比较。如果在这种情况下使用了一般的 t 检验，alpha 值将可以应用于每一次比较，所以判断错误的机会会随着比较次数的增加而增加。多元比较过程就是要给任意一次比较被错误地判为差异显著的概率设置一个上限。

输出 c 包含一个五列矩阵形式的检验结果。矩阵的每一行代表一个配对组的一次检验，行的入口显示了进行比较的均值、均值差异估计和差异的显著性水平。

例如，假设某行包含下面的入口：

```
2.0000 5.0000 1.9442 8.2206 14.4971
```

这些数字表示第 2 组均值减去第 5 组均值的结果估计值为 8.2206，真实均值的 95%置信

区间为[1.9442 14.4971]。

在本例中，置信区间不包含 0。所以在 0.05 的水平上，差异显著。若置信区间包含 0，则差异不会在 0.05 的水平上显著。

multcompare 函数还显示一个图形，图形中用一个标记和标记周围的一个区间代表每一组均值及其范围。若两个均值的区间不相连，则说明它们之间有显著差异。如果它们的区间重叠，则说明没有显著差异。可以用鼠标选择任意分组，图形中将亮显所有与它有显著差异的分组。

- c = multcompare(stats, alpha) 确定图形和 c 矩阵中区间的置信水平。置信水平为 100*(1-alpha)%，默认时 alpha=0.05。
- c = multcompare(stats, alpha, 'displayopt') 当'displayopt'设置为'on'时，激活图形显示（默认选项），当设置为"off"时，则取消显示。
- c = multcompare(stats, alpha, 'displayopt', 'ctypze') 对于多元比较，指定临界值。它们可以是表 9-7 中所示字符串中的任意一个。

表 9-7　临界值取值表

ctype 取值	含　义
'hsd'	使用 Tukey 的显著性差异准则。它是默认选项，且基于学生化极差分布。对于平衡单因素方差分析和相同大小样本的近似过程，该方法是最优选择。对于样本大小不同的单因素方差分析，该法已被证明是保守的方法。根据未获证明的 Tukey-Kramer 猜想，它对于进行比较的相关数量问题还是精确的，就像在不平衡协变量值方差分析中一样
'lsd'	使用 Tukey 的最小显著差异过程。该过程是一个简单的 t 检验。如果备择检验（即单因素方差分析的 F 统计量）结果说明差异显著，则该法是合理的。若无条件使用该方法，它将不能保证多元方差分析的结果是可靠的
'bonferroni'	使用源于 t 分布的临界值，经过 Bonferroni 调整以后，对多元比较进行补充。本过程比较保守，但比 Scheffe 过程要好一些
'dunn-sidak'	对多元比较进行调整以后使用源于 t 分布的临界值。调整的方法由 Dunn 提出，并被 Sidak 证明是正确的。该过程与 Bonferroni 过程的效果接近，但没有它保守
'scheffe'	使用源于从 F 分布得到的源于'scheffe'的 S 过程的临界值。该过程提供一个对所有均值线性组合进行比较的模拟置信水平。对于配对样本的简单比较，该方法是保守的

- c = multcompare(stats, alpha, 'displayopt', 'ctype', 'estimate') 指定进行比较的估计。估计的允许值取决于 stats 结果的来源函数，参见表 9-8。

表 9-8　估计允许值表

来　源	估计的允许取值
'anova1'	忽略。总是比较组均值
'anova2'	用'column'(默认选项)或'row'比较列或行均值
'anovan'	忽略。总是比较总体的边缘均值，就像 dim 变量指定的一样
'aoctool'	用'slope'、'intercept'或'pmm'比较斜率、截距或总体边缘均值。若方差分析模型不包括斜率，则不能用'slope'参数。若不包括截距，则不可能比较
'friedman'	忽略。总是比较列秩的均值
'kruskalwallis'	忽略。总是比较组秩的均值

● c = multcompare(stats, alpha, 'displayopt', 'ctype', 'estimate', dim) 指定进行比较的总体边缘均值。只有当 stats 结构由 anovan 函数创建时，该变量才能使用。对于 n 个因子的 n 因素方差分析，可以指定 dim 作为一个标量或 1 和 n 之间的整数向量。默认值为 1。例如，如果 dim = 1，则进行比较的估计为第 1 个分组变量的每个值的均值剔除其他分组变量的效应得到的结果，就像设计被平衡了一样。若 dim = [1 3]，则总体边缘均值用第 1 个和第 3 个分组变量的组合进行计算，剔除第 2 个分组变量的效应。若需要拟合一个奇异模型，则有些单元均值可能得不到估计值，且所有依赖于这些单元均值的总体边缘均值将被赋值为 NaN（即空值）。

Milliken 和 Johnson (1992)，Searle，Speed 和 Milliken (1980)分别对总体边缘均值进行了描述。总体边缘均值的实质是通过固定由 dim 指定的因子的值来剔除不平衡设计的所有效应，并剔除其他因子的效应，就像每个因子的组合次数相等。总体边缘均值的定义与每个因子组合上的观测值个数无关。

● [c, m] = multcompare(...) 返回一个附加矩阵 m。m 的第 1 列包含每个组均值的估计值（或任何正在比较的统计量）。第 2 列包含它们的标准差。

● [c, m, h] = multcompare(...) 对于比较图形，返回一个句柄 h。注意，图形的标题包含图形作用的建议，X 轴标签包含的信息为哪些均值与所选择的均值有显著差异。若你计划使用本图，并希望忽略标题和 X 轴标签的显示，可以使用图形窗口的交互特性或用下面的命令来删除它们：

```
title(' ')
xlabel(' ')
```

2. 应用实例

【例 9-22】 前面我们曾经使用结构梁的强度数据进行了单因素方差分析，得出 3 种不同的梁具有不同强度的结论。现在确定它们的差异在哪里。首先，我们创建一个数据数组，并进行单因素方差分析：

```
strength = [82  86  79  83  84  85  86  87  74  82  78  75  76  77  79 ...
               79  77  78  82  79];
alloy = {'st', 'st', 'st' ,'st' ,'st' ,'st' ,'st' ,'st' ,...
          'al1' ,'al1' ,'al1' ,'al1' ,'al1' ,'al1' ,...
          'al2' ,'al2' ,'al2' ,'al2' ,'al2' ,'al2'};
[p,a,s] = anova1(strength, alloy);
```

生成方差分析表和箱形图如图 9-15 和图 9-16 所示。

Figure No. 1: One-way ANOVA

File Edit View Insert Tools Window Help

ANOVA Table

Source	SS	df	MS	F	Prob>F
Groups	184.8	2	92.4	15.4	0.0002
Error	102	17	6		
Total	286.8	19			

图 9-15　方差分析表

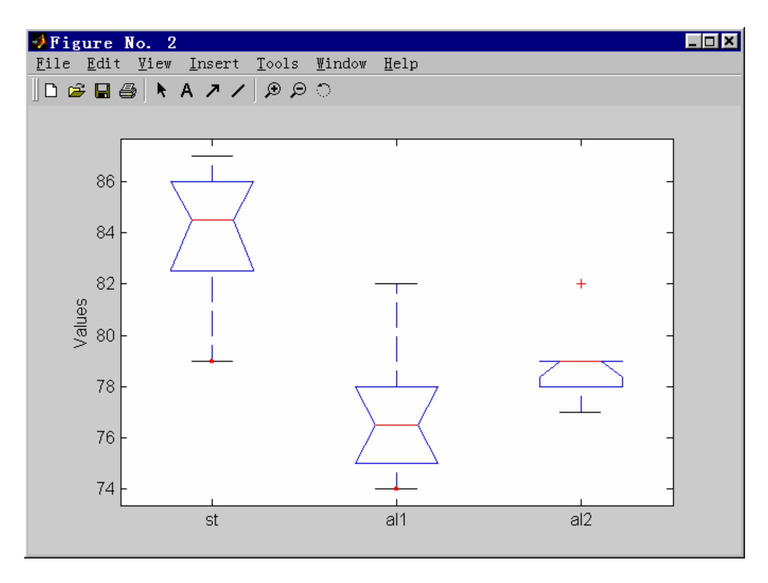


图 9-16　箱形图

输出 s 将作为后面多元比较分析的输入：

```
multcompare(s)
ans =
   1.0000    2.0000    3.6064     7.0000    10.3936
   1.0000    3.0000    1.6064     5.0000     8.3936
   2.0000    3.0000   -5.6280    -2.0000     1.6280
```

输出矩阵的第 3 行显示两种强度之间的差异不是很显著。差异的 95%置信区间为[-5.6 1.6]，所以我们不能拒绝真实差异为 0 的零假设。

前两行显示两个与第 1 组（steel）有关的比较的置信区间不包含 0。换句话说，它们的差异是显著的。图 9-16 中显示了同样的信息。

第 10 章　统计过程控制

统计过程控制（SPC）是达到和监督产品质量的许多方法的一个集体术语。这些方法比较简单，使得它们即使在生产环境中也易于操作。

MATLAB 中提供了过程控制图和过程性能图两类图形。其中过程控制图包括指数加权移动平均图、标准离差控制图和 X 条图。过程性能图包括过程性能指数图、过程概率密度图和直方图等。

10.1　过程控制图

10.1.1　基本原理

过程控制图是对产品进行质量控制的有效手段。在控制图中，用直线或折线表示质量控制的上限和下限。按照正常的工序，产品质量曲线应该在控制上限和控制下限之间摆动。如果产品质量曲线在某些部位超出了控制上限和控制下限的范围，则表示在对应的生产环节出现了问题。这些问题有可能是原材料的问题，也可能是操作人员的问题，或者是由于其他问题。总之，控制图中显示出了问题的存在，生产管理者就可以有的放矢地分析问题、解决问题，保证生产的顺利进行。

常见的过程控制图有均值控制图、标准差控制图和指数加权移动平均控制图等。在 MATLAB 中分别用 xbarplot 函数、schart 函数和 ewmaplot 函数实现。

10.1.2　有关函数介绍

1．ewmaplot 函数

利用该函数生成指数加权移动平均图（EWMA），便于统计过程控制（SPC）。

● ewmaplot(data) 生成 data 中分组响应的指数加权移动平均图。data 的行包含给定时间上的重复观测值。各行应按时间先后排序。

● ewmaplot(data, lambda) 生成 data 中分组响应的指数加权移动平均图，并指定当前预测被以往观测值影响的程度。lambda 越大，赋给过去观测值的权重也越大。默认时，lambda = 0.4；lambda 必须在 0 和 1 之间。

● ewmaplot(data, lambda, alpha) 生成 data 中分组响应的指数加权移动平均图，并指定绘图时需要的置信上限和置信下限。默认时，alpha 为 0.0027。该值生成 3 倍σ界限。

```
norminv(1-0.0027/2)
ans =
      3
```

用表达式 2*(1-normcdf(k))，可以得到 k 倍σ界限。例如，对于 2 倍σ界限，正确的 alpha 值为 0.0455, 就像下面所显示的：

```
k = 2;
```

```
2*(1-normcdf(k))
ans =
      0.0455
```

● ewmaplot(data, lambda, alpha, specs) 生成 data 中分组响应的指数加权移动平均图，并指定一个二元素向量 specs，以指定响应的上界和下界。

● h = ewmaplot(...) 返回句柄向量到图中直线。

【例 10-1】 对于具有缓慢移动均值的过程，指数加权移动平均图比 x 条图更能监测它的过程。下面的模拟演示了一个缓慢线性漂移均值的指数加权移动平均图。

```
t = (1:28)';
r = normrnd(10+0.02*t(:,ones(4,1)),0.5);
ewmaplot(r,0.4,0.01,[9.75 10.75])
```

生成图如图 10-1 所示。

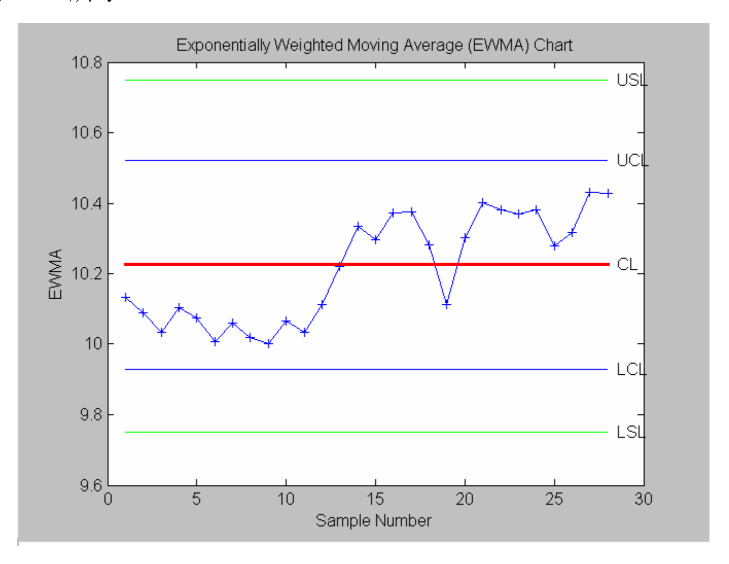


图 10-1　指数加权移动平均图

第 28 组的指数加权移动平均值比随机期望值要高。如果我们对这个过程进行了连续的监测，可能会发现该漂移，并进一步调查产生该漂移的原因。

2．schart 函数

利用该函数创建标准离差控制图，其调用格式为：

● schart(data) 显示 data 中分组响应的 S 图。data 的行包含给定时间的重复观测值。各行必须按时间排序。图中包含每个组的样本标准离差 s、平均 s 值处的中心线和控制界限的上限和下限。该界限位于中心线两侧 3 倍σ 距离处，其中σ 为 s 的标准离差的估计值。若该过程可控，则少于千分之三的观测值将会随机地落在控制界限以外。所以，如果观测点在界限以外，则说明过程失控。

● schart(data, conf) 允许在绘制控制界限时控制置信水平的上界和下界。默认时，conf = 0.9973，生成 3 倍σ 界限。

```
norminv(1 - (1-.9973)/2)
ans =
          3
```

使用表达式 1-2*(1-normcdf(k))，可以得到 k 倍σ 界限。例如，对于 2 倍σ 界限，正确的 conf 值为 0.9545，如下所示：

```
k = 2;
1-2*(1-normcdf(k))
ans =
       0.9545
```

● schart(data, conf, specs) 根据二元素向量 specs 中指定的界限绘图。

● [outliers, h] = schart(data, conf, specs) 当 data 的均值失控时，返回异常值（用指数向量表示）到各行，并返回图中直线的句柄向量 h。

【例 10-2】 本例绘制新机器部件测量值的 S 图。这些测量值在 36 小时中每隔 1 小时取一次。runout 矩阵包含随机挑选的 4 个部件的测量值。这些值显示，部件实测半径与目标半径之间相差千分之一英寸。

```
load parts
schart(runout)
```

生成的图形如图 10-2 所示。

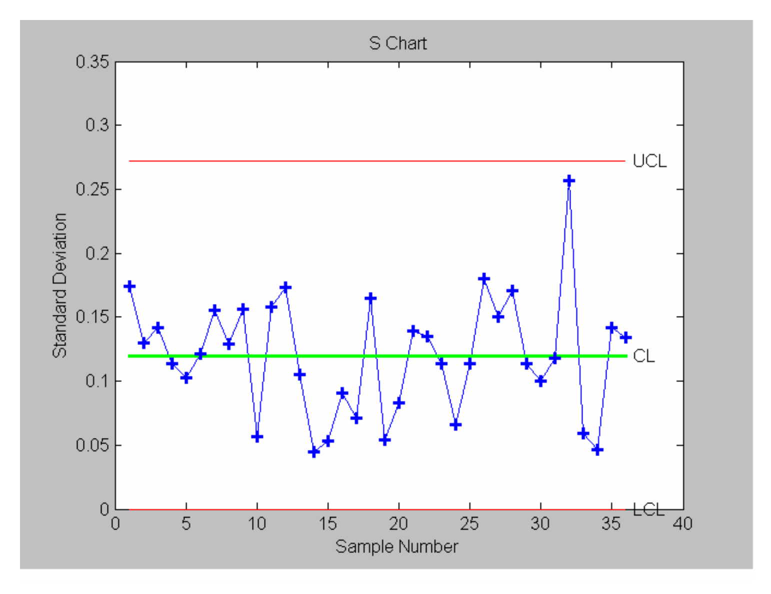


图 10-2　标准离差控制图

因为所有的点都在控制界限以内，所以各次级分组内的随机变异是连贯的，没有证据表明过程失控。

3. xbarplot 函数

利用该函数生成 X 条图。调用方法如下：

● xbarplot(DATA) 显示 DATA 中分组响应的 x 条图。DATA 的行包含给定时间的重复观测值。各行必须按时间排序。图中包含每个组的样本均值、均值处的中心线和控制界限的上限和下限。该界限位于中心线两侧 3 倍σ距离处，其中σ为样本均值的标准离差的估计值。若该过程可控，则少于千分之三的观测值将会随机地落在控制界限以外。所以，如果观测点在界限以外，则说明过程失控。

● xbarplot(DATA, conf) 允许在绘制控制界限时控制置信水平的上界和下界。默认时，conf = 0.9973，生成 3 倍σ界限。

```
norminv(1-(1-.9973)/2)
ans =
        3
```

使用表达式 1-2*(1-normcdf(k))，可以得到 k 倍σ界限。例如，对于 2 倍σ界限，正确的 conf 值为 0.9545，如下所示：

```
k = 2;
1-2*(1-normcdf(k))
ans =
       0.9545
```

● xbarplot(DATA, conf, specs) 根据二元素向量 specs 中指定的界限绘图。

● xbarplot(DATA, conf, specs, 'sigmaest') 指定 X 条图如何估计标准差。可有下面一些选项：

's'——默认时，使用各组标准差的均值；

'v'——使用合并方差估计的平方根；

'r'——使用每个组的平均极差，每组需要 25 个或更少的观测值。

● [outlier, h] = xbarplot(DATA, conf, specs) 当 DATA 的均值失控时，返回异常值（用指数向量表示）到各行，并返回图中直线的句柄向量（h）。

【例 10-3】 本例绘制新机器部件测量值的 x 条图。这些测量值在 36 小时中每隔 1 小时取一次。runout 矩阵包含随机挑选的 4 个部件的测量值。这些值显示，部件实测半径与目标半径之间相差千分之一英寸。

```
load parts
xbarplot(runout,0.999,[-0.5 0.5])
```

从生成的图 10-3 中可以看出，第 21 组和第 25 组中的点失控，所以这些组中的均值比期望值高。说明该过程有些失控，需要查明原因，采取对策。

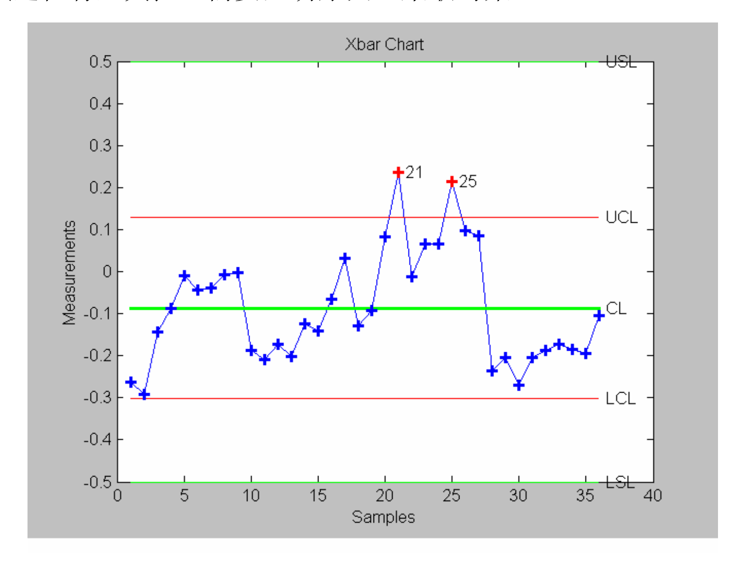


图 10-3　X 条图

10.2 过程性能图

在进行全面生产之前，许多厂家首先要进行试验性研究，看一定的工序能否达到工程计划指定的生产要求。利用统计模型得到的能力研究数据，可以获得不能完成指定计划部分的百分比。

常见的过程性能图有过程概率图、叠加了正态密度曲线的直方图、指定区间内的正态密度曲线图等。在 MATLAB 中分别用 capable 函数、capaplot 函数、histfit 函数和 normspec 函数实现。

1．capable 函数

利用该函数计算过程性能指数，其调用方式如下：

● capable(data, lower, upper) 计算样本 data 中的数据落在 lower 和 upper 指定的范围之外的可能性。前提假设是 data 向量中的度量值服从均值和方差都为常数的正态分布，并且测量在统计上是独立的。

● [p, Cp, Cpk] = capable(data, lower, upper) 还返回可能性指数 Cp（C_p）和 Cpk（C_{pk}）。C_p 是指定数据的极差与过程标准误差估计的 6 倍之间的比值，即

$$C_p = \frac{USL - LSL}{6\sigma}$$

对于目标处有均值的过程，C_p=1 转换为略大于千分之一。近来，很多工业部门将产品质量目标缺陷标准设置为百万分之一，与之对应的 $C_p = 1.6$，C_p 越高，则生产过程的性能越好。

$$C_{pk} = \min(\frac{USL - \mu}{3\sigma}, \frac{\mu - LSL}{3\sigma})$$

C_{pk}是过程均值与指定上界或下界之间的差值和3倍过程标准差的估计值之间的比率. 其中 μ 为过程均值。对于过程均值没有保持在目标上的情况，C_{pk}更能描述过程的性能。

【例 10-4】 假设要求某机器部件的大小误差限制在千分之三英寸以内，且其误差服从正态分布。如果部件生产过程中平均厚了千分之一英寸，并且标准差为千分之一英寸。求该过程的性能指数是多少？

```
data = normrnd(1,1,30,1);
[p,Cp,Cpk] = capable(data,[-3 3]);
indices = [p Cp Cpk]
indices =
     0.0172    1.1144    0.7053
```

结果发现有千分之十七的部件超出了指定要求。因为过程较分散，所以 C_{pk} 比 C_p 小。

2．capaplot 函数

利用该函数使用过程概率图。调用格式如下：

● capaplot(data, specs) 假设向量数据中的观测量服从均值和方差未知的正态分布。对其进行拟合并用图形显示某新观测量的 T 分布。分布中上下界限之间的部分（包含在二元素向量 specs 中）在图中是隐藏的。

● [p, h] = capaplot(data, specs) 返回新观测量在评估范围内的概率 p 和图形元素的句柄 h。

【例 10-5】 继续使用前面的例子。

```
data = normrnd(1,1,30,1);
```

```
p = capaplot(data,[-3 3])
p =
    0.9784
```

新观测量的概率 specs 为 97.84%。生成的图形如图 10-4 所示。

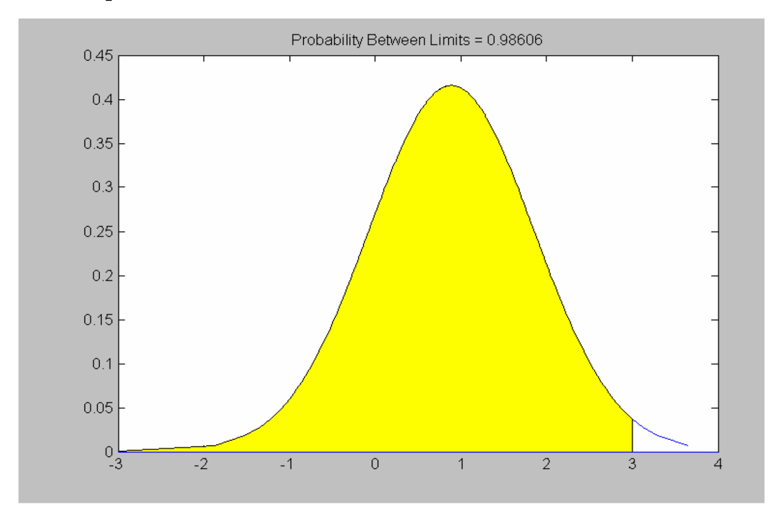


图 10-4　过程概率图

3．histfit 函数

利用该函数创建添加了正态密度的直方图，其调用格式为：

● histfit(data, nbins)　生成 data 数据的直方图，其中有 nbins 个条形。当只有一个输入变量时，nbins 设置为 data 中元素个数的平方根。

● h = histfit(data, nbins)　返回图中线条的句柄向量。h(1)为直方图的句柄，h(2)为密度曲线的句柄。

【例 10-6】

```
r = normrnd(10,1,100,1);
histfit(r)
```

生成的图形如图 10-5 所示。

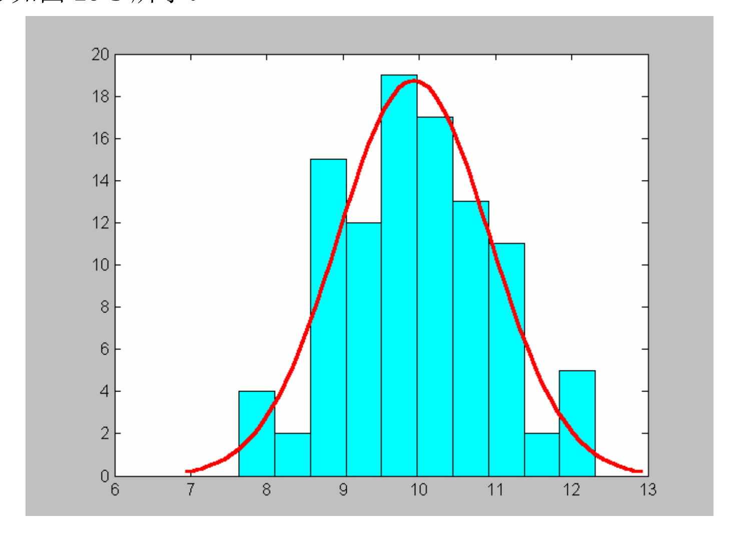


图 10-5　正态密度叠加直方图

4. normspec 函数

利用该函数绘制指定区间内的正态密度曲线，其调用格式如下：

- p = normspec(specs, mu, sigma) 绘制由向量 specs 中两元素确定的下界和上界之间的正态密度曲线，mu 和 sigma 为正态分布的参数。
- [p, h] = normspec(specs, mu, sigma) 返回样本数据落在下限与上限之间的概率 p。h 为直线对象的句柄。若 specs(1)= -Inf（-∞），则没有下限；同样，若 specs(2) = Inf，则没有上限。

【例 10-7】 假设厂家生产 10 盎司一箱的玉米片。在装箱过程中实际装入的玉米片的重量与期望重量之间有一定误差。如果每箱玉米片的平均重量为 11.5 盎司，标准差为 1.25 盎司，问玉米片重量大于 10 盎司的装箱数占多少百分比？

```
normspec([10 Inf],11.5,1.25)
```

从图 10-6 中可以看出，玉米片重量大于 10 盎司的装箱数占 88.5%。

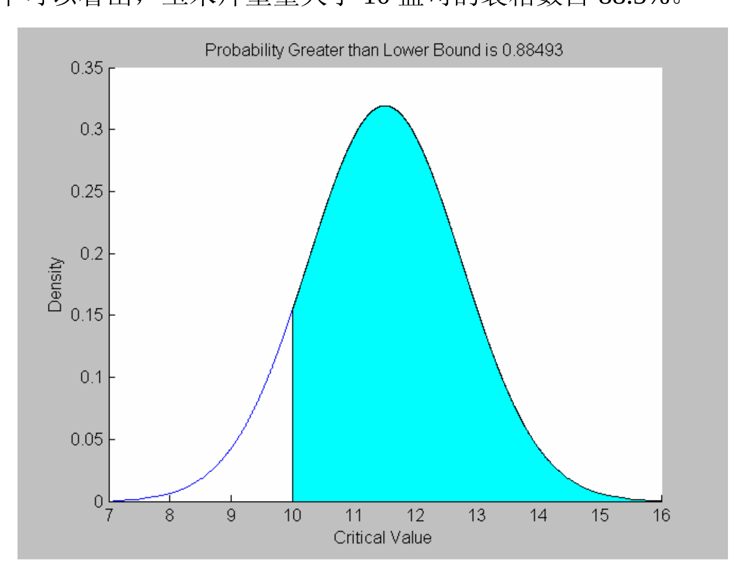


图 10-6　指定区间的正态密度曲线

第 11 章　试 验 设 计

试验设计是对可能的试验方案进行优化选择的科学，它研究如何科学合理地安排试验因素，研究试验中各影响因素之间的关系。试验设计的目的就是确定如何收集数据，以供统计推断之用。好的设计，可以收到事半功倍的效果。MATLAB 中介绍了完全析因设计、不完全析因设计、响应面设计和 D-优化设计等四种试验设计方法。

11.1　完全析因设计

11.1.1　基本原理

假设现在要确定加工过程中的差异性是由切削部件的机床引起的，还是由操作该机床的操作员引起的。如果总是由同一个操作员操作同一台机床，则不能确定问题是由操作员还是由机床引起的。但是，如果允许每个操作员都去操作每一台机床，则可以区分他们之间的差别。这样一个过程，就是析因过程。

假设现在有 4 个操作员和 3 台机床，则其析因设计为：

```
d = fullfact([4 3])
d =
     1     1
     2     1
     3     1
     4     1
     1     2
     2     2
     3     2
     4     2
     1     3
     2     3
     3     3
     4     3
```

d 的每一行代表一个操作员/机床组合。上面有 4*3 = 12 行。

析因设计的一个特例是所有的变量都只有两个值。假设要很快地确定某个过程中 3 个变量的值高或低，可以做如下设计：

```
d2 = ff2n(3)
d2 =
     0     0     0
     0     0     1
     0     1     0
     0     1     1
     1     0     0
     1     0     1
     1     1     0
     1     1     1
```

共有 $2^3 = 8$ 个组合。

11.1.2　有关函数介绍

1．ff2n 函数

利用该函数进行二水平完全析因设计，其调用格式如下：

- X = ff2n(n) 创建一个二水平完全析因设计 X。n 为 X 的列数，行数为 2n。

【例 11-1】

```
X = ff2n(3)
X =
     0     0     0
     0     0     1
     0     1     0
     0     1     1
     1     0     0
     1     0     1
     1     1     0
     1     1     1
```

X 表示 0 到 $2n-1$ 的数字。

2．fullfact 函数

利用该函数进行完全析因试验设计，其调用格式如下：

- design = fullfact(levels) 给定因子设置，进行完全析因设计。levels 向量中的每一个元素指定 design 对应列中惟一元素的个数。例如，若 levels 的第 1 个元素是 3，则 design 的第 1 列只包含从 1 到 3 的整数。

【例 11-2】　若 levels = [2 4]，则 fullfact 函数生成一个 8 次的设计，其中第 1 列中有 2 个水平，第 2 列中有 4 个水平。

```
d = fullfact([2 4])
d =
     1     1
     2     1
     1     2
     2     2
     1     3
     2     3
     1     4
     2     4
```

11.2　不完全析因设计

11.2.1　基本数学原理

析因设计的困难之一是当变量增加时，进行析因设计的组合将呈指数增长。例如，在上面的例子中，如果变量个数不是 3，而是 7，按照完全析因设计需要试验 $2^7 = 128$ 次。

如果系统中各变量间没有协同作用，则需要的试验次数将大大减少。上例中，理论上的最小试验次数为 8 次。使用 hadamard 函数可以定义最小试验次数。

```
X = hadamard(8)
X =
     1     1     1     1     1     1     1     1
     1    -1     1    -1     1    -1     1    -1
     1     1    -1    -1     1     1    -1    -1
     1    -1    -1     1     1    -1    -1     1
     1     1     1     1    -1    -1    -1    -1
     1    -1     1    -1    -1     1    -1     1
     1     1    -1    -1    -1    -1     1     1
     1    -1    -1     1    -1     1     1    -1
```

最后 7 列是真实的变量集（-1 表示低值，1 表示高值）。利用第 1 列可以衡量均值在线性方程$Y = X\beta + \varepsilon$中的影响。

利用 Plackett-Burman 设计，可以用较少的试验次数研究每个变量的主效应。对于上面的例子，使用不完全析因设计，试验次数只是完全析因设计的 8/128。不完全析因设计的缺点是，如果一个变量的效应随另一个变量的效应变化，则估计的效应会出现偏差。

11.2.2　有关函数介绍

利用 fracfact 函数生成源于生成器的不完全析因设计，其调用格式如下：

● x = fracfact('gen') 根据生成器字符串 gen 指定的内容生成不完全析因设计，并返回设计点的矩阵 x。输入字符串 gen 是一个生成器字符串，由被空格间隔的"words"组成。每个单词描述输出设计列是如何由完全因子的列组成的。一种典型的情况是，gen 将为前面几个因子包含单字母单词，另外一些多字母单词描述剩下的因子是如何由前面几个所组成的。

输出矩阵 ***X*** 是一个二水平完全析因设计的一部分。假设在 gen 中有 *m* 个单词，且每个单词由字母表中的前 *n* 个字母的子集组成。输出矩阵 ***X*** 有 2*n* 行、*m* 列。令 F 代表二水平完全析因设计，就像被 ff2n(n)函数生成的一样。

● [x,conf] = fracfact('gen') 还返回一个单元数组 conf，它描述主效应的组成模式和所有的二因子交互作用。

11.2.3　应用实例

【例 11-3】 如果希望进行一个试验，研究与某响应有关的 4 个因子的效应，但现在由于条件限制，只能进行 8 次试验（一个 run 是试验在指定因子组合上的简单重复）。试验的目的是确定哪些因子对效应有影响。在有些因子配对之间可能会有交互作用。

进行完全析因设计需要 16 次试验，但如果假设没有 3 因子交互效应，则通过 8 次就可以估计主因子效应。

```
[x,conf] = fracfact('a b c abc')
x =
 -1    -1    -1    -1
 -1    -1     1     1
 -1     1    -1     1
 -1     1     1    -1
  1    -1    -1     1
  1    -1     1    -1
  1     1    -1    -1
  1     1     1     1
```

```
conf =
    'Term'     'Generator'    'Confounding'
    'X1'       'a'            'X1'
    'X2'       'b'            'X2'
    'X3'       'c'            'X3'
    'X4'       'abc'          'X4'
    'X1*X2'    'ab'           'X1*X2+X3*X4'
    'X1*X3'    'ac'           'X1*X3+X2*X4'
    'X1*X4'    'bc'           'X1*X4+X2*X3'
    'X2*X3'    'bc'           'X1*X4+X2*X3'
    'X2*X4'    'ac'           'X1*X3+X2*X4'
    'X3*X4'    'ab'           'X1*X2+X3*X4'
```

X 矩阵的前面 3 列组成了一个完全析因设计。最后一列通过乘以后面 3 列来得到。组成模式表明，对于所有 4 个因子而言，其主效应是可以估计的，但二因子交互效应却不能。

进行试验以后，发现'ab'效应是显著的。为了确定该效应是否来自 X1*X2 或 X3*X4，需要进行剩下的 8 次试验。通过改变最终生成器的符号可以获得它们。

```
fracfact('a b c -abc')
ans =
    -1    -1    -1     1
    -1    -1     1    -1
    -1     1    -1    -1
    -1     1     1     1
     1    -1    -1    -1
     1    -1     1     1
     1     1    -1     1
     1     1     1    -1
```

【例 11-4】 假设现在我们要研究 8 个因子的效应，进行完全析因设计需要 256 次试验，通过生成器的智能选择，发现只要 16 次就可以估计这 8 个效应，其中没有组成二因子效应。

```
[x,c] = fracfact('a b c d abc acd abd bcd');
c(1:10,:)
ans =
    'Term'     'Generator'    'Confounding'
    'X1'       'a'            'X1'
    'X2'       'b'            'X2'
    'X3'       'c'            'X3'
    'X4'       'd'            'X4'
    'X5'       'abc'          'X5'
    'X6'       'acd'          'X6'
    'X7'       'abd'          'X7'
    'X8'       'bcd'          'X8'
    'X1*X2'    'ab'           'X1*X2+X3*X5+X4*X7+X6*X8'
```

这种组成模式显示主效应不是由二因子交互效应组成的。最后一行显示一组含有 4 个二因子的交互效应已经组成了。生成器其他选择不需要相同的特性。

```
[x,c] = fracfact('a b c d ab cd ad bc');
c(1:10,:)
ans =
    'Term'     'Generator'    'Confounding'
```

```
'X1'      'a'        'X1+X2*X5+X4*X7'
'X2'      'b'        'X2+X1*X5+X3*X8'
'X3'      'c'        'X3+X2*X8+X4*X6'
'X4'      'd'        'X4+X1*X7+X3*X6'
'X5'      'ab'       'X5+X1*X2'
'X6'      'cd'       'X6+X3*X4'
'X7'      'ad'       'X7+X1*X4'
'X8'      'bc'       'X8+X2*X3'
'X1*X2'   'ab'       'X5+X1*X2'
```

这里，所有的主效应由一个或多个二因子交互效应组成。

11.3　响应面设计

有时候，仅使用简单的线性和交互模型是不够的，比如，假设输出是缺陷或产出，目标是使缺陷最小化，或产出最大化。如果最优点在进行试验的区域内部，就需要一个能表示曲率的数学模型。最简单的模型为二次模型，所有因子都具有线性项和二次项，另外还具有所有因子之间的两两交互项。拟合这些类型模型的设计称为响应面设计。此类设计之一是完全析因设计，每个输入有 3 个值。尽管统计工具箱可以进行此项设计，但是在大多数情况下，它不是一个令人满意的设计。因为它往往比正常情况下拟合模型时需要更多的运算。在响应面模型设计中最常用的两个设计是中心合成设计和 Box-Behnken 设计。在这些设计中，输入为 3 个和 5 个不同的值（或水平），但不是这些值的所有组合都出现在设计中。这里介绍的函数创建指定的响应面设计：中心合成设计和 Box-Behnken 设计。如果这些还不能满足需要，则考虑创建一个 D-优化设计。

1．中心合成设计

中心合成设计是可以拟合二次完全析因设计的响应面设计。要图形显示一个中心合成设计，想象你有一些在低值和高值之间变化的因子，为了简便，假设每个因子在-1 和 +1 之间变化。一个中心合成设计由单位立方体角点上的点、轴上或立方体外部的星点和原点处的中心点等三部分组成。中心合成设计有 3 类，外接（CCC）设计与上面描述的相同。内接（CCI）设计也与上面描述的相同，但进行了标准化，使得星点的值位于-1 和 +1 之间，而且立方体点位于立方体的内部。表面（CCF）设计中，星点位于立方体的面上，每个因子具有 3 个水平，而其他设计中每个因子具有五个水平。图 11-1 显示了三因子的 3 种类型的设计，其中，图(a)为 CCC 设计，图(b)为 CCI 设计，图(c)为 CCF 设计。

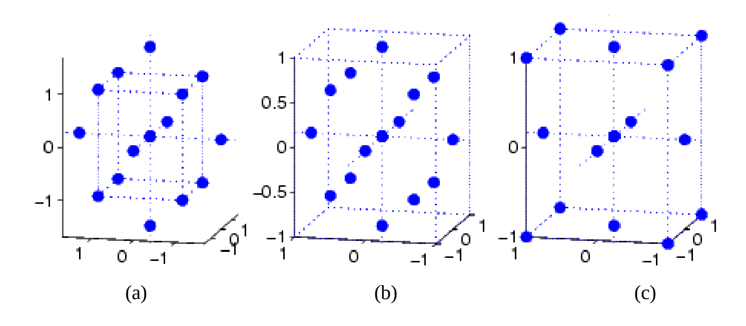


图 11-1　三因子中心合成设计

2．Box-Behnken 设计

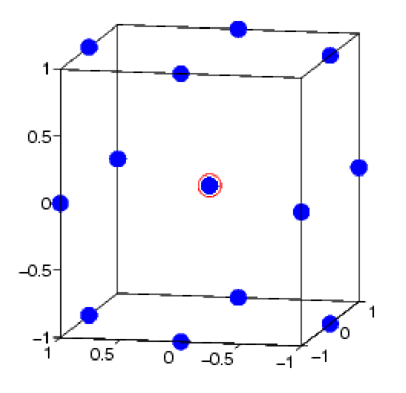


图 11-2　三因子的 Box-Behnken 设计

与中心合成设计一样，Box-Behnken 设计是可以拟合一个二次完全析因设计的响应面设计。与大部分中心合成设计方法不同的是，Box-Behnken 设计对每个因子使用三水平设计，这在样本较小而且因子为定量数据时很有吸引力。中心合成表面设计也使用三因子水平设计。但是，它不象 Box-Behnken 设计那样可以旋转。另一方面，Box-Behnken 设计被认为对于立方体角点上值的预测能力较弱，因为与 CCF 不同，它们在立方体角点上没有点。图 11-2 显示一个三因子的 Box-Behnken 设计。圆心显示在原点，并且可以重复运算多次。

在 MATLAB 统计工具箱中，用 ccdesign 函数进行中心合成设计，用 bbdesign 函数进行 Box-Behnken 设计。

11.4　D-优化设计

11.4.1　基本数学原理

前面的试验设计方法用于 20 世纪早期，从 20 世纪 70 年代开始，统计学家们开始应用计算机技术进行优化意义上的试验设计。D-优化设计使 Fisher 信息矩阵 $\boldsymbol{X}^T\boldsymbol{X}$ 的行列式最大化。该矩阵与参数的协方差矩阵的逆成比例。所以，使 $\det(\boldsymbol{X}^T\boldsymbol{X})$最大化等价于使参数协方差矩阵的行列式最小化。D-优化设计使线性模型参数 β 的回归估计的置信椭球的体积最小化。

利用 cordexch 函数和 rowexch 函数计算给定模型的 D-优化设计。这两个函数将在后面详细介绍，他们都使用了迭代算法，通过变化增量来改进初始设计。在坐标交换算法中（对于 cordexch 函数），增量为设计矩阵的单个元素；在行交换算法中（对于 rowexch 函数），增量为设计矩阵的行的元素。

11.4.2　有关函数介绍

1．cordexch 函数

该函数提供进行 D-优化设计的坐标交换算法，其调用格式为：

● settings = cordexch(nfactors,n 次) 生成因子设置矩阵 **settings**，对于 D-优化设计使用一个带常数项的线性附加模型。**settings** 参数有 n 次行、nfactors 列。

● [settings, X] = cordexch(nfactors, n 次) 还生成相关的设计矩阵 X。

● [settings, X] = cordexch(nfactors, n 次, 'model') 为了拟合一个指定的回归模型进行设计。输入参数'model'可以是下面参数中的一个：

'interaction'——包括常数项、线性项和交互项；

'quadratic'——交互项加上平方项；

'purequadratic'——包括常数项、线性项和平方项。

【例 11-5】 下面用二次模型进行 2 因子 9 次 D-优化设计。

```
settings = cordexch(2,9,'quadratic')
settings =
```

```
-1     1
 1     1
 0     1
 1    -1
-1    -1
 0    -1
 1     0
 0     0
-1     0
```

2．daugment 函数

利用该函数实现试验设计的 D-优化扩展（增广），其调用格式为：

- settings = daugment(startdes, n) 扩展一个初始试验设计 startdes，进行 n 次新的测试。
- [settings, X] = daugment(startdes, n, 'model') 还提供设计矩阵 ***X***。输入'model'控制回归模型的阶次。默认时，daugment 函数设定一个线性附加模型。另外，'model'可以是下面字符串中的一种：

'interaction'——包括常数项、线性项和交互项；

'quadratic'——交互项加上平方项；

'purequadratic'——包括常数项、线性项和平方项。

daugment 函数使用的是坐标转换算法。

实际上，试验是一个迭代过程。我们经常希望在全面试验的基础上增加试验次数，以更深入全面地了解系统。使用函数 daugment 可以优先选择这些额外的试验。

【例 11-6】 假设为了拟合 4 输入的线性模型，我们进行了 8 次试验。

```
settings = cordexch(4,8)
settings =
  1    -1     1     1
 -1    -1     1    -1
 -1     1     1     1
  1     1     1    -1
 -1     1    -1     1
  1    -1    -1     1
 -1    -1    -1    -1
  1     1    -1    -1
```

该设计对于拟合 4 输入的线性模型已经足够了，但它不能拟合 6 个叉积项。假设现在要增加 8 次试验，以拟合这些额外的项，需要进行下面的运算：

```
[augmented, X] = daugment(settings,8,'i');
augmented
augmented =
    1      -1       1       1
   -1      -1       1      -1
   -1       1       1       1
    1       1       1      -1
   -1       1      -1       1
    1      -1      -1       1
   -1      -1      -1      -1
    1       1      -1      -1
```

```
    -1      -1      -1       1
     1       1       1       1
    -1      -1       1       1
    -1       1       1      -1
     1      -1       1      -1
     1      -1      -1      -1
    -1       1      -1      -1
     1       1      -1       1
info = X'*X
info =
    16     0     0     0     0     0     0     0     0     0     0
     0    16     0     0     0     0     0     0     0     0     0
     0     0    16     0     0     0     0     0     0     0     0
     0     0     0    16     0     0     0     0     0     0     0
     0     0     0     0    16     0     0     0     0     0     0
     0     0     0     0     0    16     0     0     0     0     0
     0     0     0     0     0     0    16     0     0     0     0
     0     0     0     0     0     0     0    16     0     0     0
     0     0     0     0     0     0     0     0    16     0     0
     0     0     0     0     0     0     0     0     0    16     0
     0     0     0     0     0     0     0     0     0     0    16
```

变量化设计是正交的，因为 ***X*'****X*** 是单位矩阵。实际上，该设计与 2^4 完全析因设计相同。

【例 11-7】 本例在 2^2 不完全析因设计的基础上添加 5 次试验，拟合二次模型。

```
startdes = [-1 -1; 1 -1; -1 1; 1 1];
settings = daugment(startdes, 5, 'quadratic')
settings =
    -1    -1
     1    -1
    -1     1
     1     1
     1     0
    -1     0
     0     1
     0     0
     0    -1
```

结果为 32 析因设计。

3．dcovary 函数

利用 dcovary 函数，用指定的协方差进行 D-优化设计，其调用格式为：

- settings = dcovary(factors,covariates,'model') 为每一次运行创建一个有固定协变量约束的 D-优化设计。factors 为希望控制的试验变量个数。
- [settings,X] = dcovary(factors,covariates,'model') 还创建相关的设计矩阵 X。输入参数‘model’控制回归模型的阶次。默认时，dcovary 函数假设为一线性附加模型。另外，‘model’可以是下面字符串中的任意一个：

'interaction'——包括常数项、线性项和交互项；

'quadratic'——交互项加上平方项；

'purequadratic'——包括常数项、线性项和平方项。

有时无法控制每一个试验输入。但事先知道有些输入的值。利用函数 dcovary 可以为每次试验选择 settings 矩阵，以使过程中除了线性飘移以外的信息最大化。

【例 11-8】 假设要将一个 8 次的试验转换为 4 个大小为 2 的区组，以拟合二因子线性模型。

```
covariates = dummyvar([1 1 2 2 3 3 4 4]);
settings = dcovary(2,covariates(:,1:3),'linear')
settings =
  1     1     1     0     0
 -1    -1     1     0     0
 -1     1     0     1     0
  1    -1     0     1     0
  1     1     0     0     1
 -1    -1     0     0     1
 -1     1     0     0     0
  1    -1     0     0     0
```

输出矩阵的前两列包含这两个因子的设置。

4．hadamard 函数

利用该函数生成 Hadamard 矩阵，其调用格式为：

● H = hadamard(n) 返回阶次为 n 的 Hadamard 矩阵。

其中，Hadamard 是由 1 和-1 组成的矩阵，其列是正交的，且有

H'*H = n*I

利用 size 函数可以得到 n，利用 eye 函数可以得到 I，即

[n n] = size(H)

I = eye(n, n)

只有当 rem(n, 4) = 0 时，存在一个 $n\times n$（$n > 2$）的 Hadamard 矩阵。该函数只有在 n，$n/12$ 或 $n/20$ 为 2 的幂次时可以控制。

【例 11-9】 命令 hadamard(4)生成一个 4×4 的矩阵。

```
1     1   1     1
1    -1   1    -1
1     1  -1    -1
1    -1  -1     1
```

5．rowexch 函数

该函数提供进行 D-优化设计的行交换算法，其调用格式为：

● settings = rowexch(nfactors, n) 生成因子设置矩阵 settings，用带常数项的线性累加模型进行 D-优化设计。settints 矩阵有 n 次行，nfactors 列。

● [settings, X] = rowexch(nfactors, n) 也生成相关设计矩阵 X。

● [settings, X] = rowexch(nfactors, n, 'model') 为拟合指定的回归模型生成设计。输入参数‘model’可以是下列字符串之一：

'interaction' ——包括常数项、线性项和交互项；

'quadratic' ——交互项加上平方项；

'purequadratic' ——包括常数项、线性项和平方项。

【例 11-10】 本例用交互模型演示 8 次三因子的 D-优化设计，为两水平的完全析因设计。

```
s = rowexch(3,8,'interaction')
s =
```

```
  -1       -1        1
   1       -1       -1
   1       -1        1
  -1       -1       -1
  -1        1        1
   1        1        1
  -1        1       -1
   1        1       -1
```

11.4.3　综合实例

【例 11-11】　考虑二输入的二次模型，使用坐标交换算法进行试验设计，模型的形式为

$$y = \beta_0 + \beta_1 x_1 + \beta_1 x_2 + \beta_{12} x_1 x_2 + \beta_{12} x_1^2 + \beta_{12} x_2^2 + \varepsilon$$

假设希望该 D-优化设计通过 9 次试验来拟合模型。

```
settings = cordexch(2,9,'q')
settings =
  -1     1
   1     1
   0     1
   1    -1
  -1    -1
   0    -1
   1     0
   0     0
  -1     0
```

可以将 settings 的列之间的关系绘成图形。

```
h = plot(settings(:,1),settings(:,2),'.');
set(gca,'Xtick',[-1 0 1])
set(gca,'Ytick',[-1 0 1])
set(h,'Markersize',20)
```

绘成的图形如图 11-3 所示。

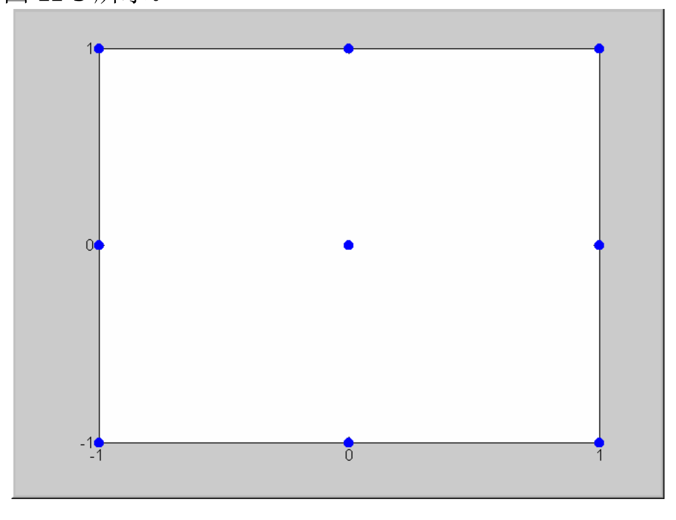


图 11-3　坐标交换算法得到的 settings 矩阵列之间的关系图

同样考虑二输入的交互模型，使用行交换算法进行试验设计。该模型的形式为

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{12} x_1 x_2 + \varepsilon$$

假设希望该 D-优化设计通过 4 次试验来拟合模型。

```
[settings, X] = rowexch(2,4,'i')
settings =
   -1     1
   -1    -1
    1    -1
    1     1
X =
    1      -1       1      -1
    1      -1      -1       1
    1       1      -1      -1
    1       1       1       1
```

settings 矩阵显示了如何改变各次试验之间的输入。X 矩阵为拟合上面回归模型的设计矩阵。X 的第 1 列拟合常数项，最后一列为第 2 列和第 3 列的积。

相关的图形表示为：

```
h = plot(settings(:,1),settings(:,2),'.');
set(gca,'Xtick',[-1 0 1])
set(gca,'Ytick',[-1 0 1])
set(h,'Markersize', 20)
```

其结果如图 11-4 所示。

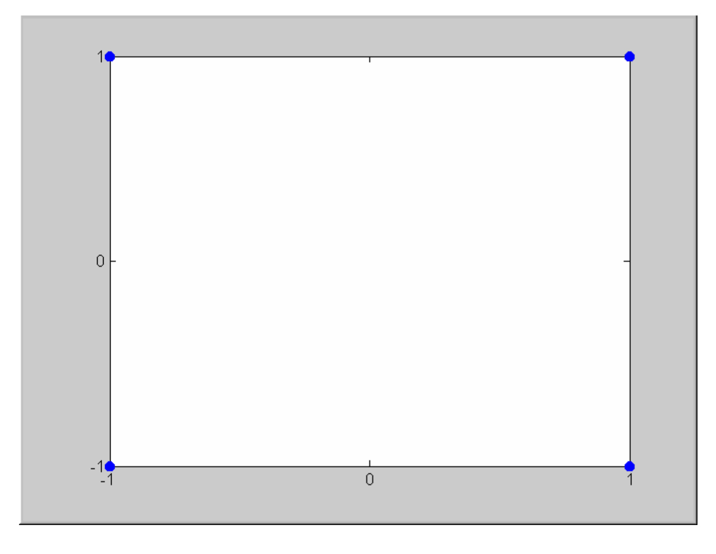


图 11-4　行交换算法得到的 settings 矩阵列之间的关系图

第 12 章　统　计　图

用图形表达样本数据的统计特征具有生动、直观的优点。MATLAB 提供了十多种常用统计图形，包括：箱形图、经验累加分布函数图、误差条图、函数交互等值线图、散点矩阵图、散点图、添加最小二乘拟合线、正态概率图、帕累托图、q-q 图、回归个案次序图、参考多项式曲线、添加参考线、交互插值等值线图、威布尔图。

12.1　箱形图

箱形图可以比较清晰地表示数据的分布特征。它由 5 个部分组成：

① 箱形上、下的横线为样本的 25%和 75%分位数，箱形顶部和底部的差值为内四分位极值；

② 箱形中间的横线为样本的中值。若该横线没在箱形中央，则说明存在偏度；

③ 箱形向上或向下延伸的直线称为“触须”，若没有异常值，样本的最大值为上触须的顶部，样本最小值为下触须的底部。默认情况下，距离箱形顶部或底部大于 1.5 倍内四分极值的值称为异常值；

④ 图中顶部的加号表示该处数据为一异常值。该值的异常可能是输入错误、测量失误或系统误差引起的；

⑤ 箱形两侧的 V 形槽口对应于样本中值的置信区间。默认情况下，箱形图没有 V 形槽口。

用 boxplot 函数生成数据样本的箱形图，其调用格式如下：

● boxplot(X)　为 X 矩阵的每一列生成一个箱形图。箱形上、下四分位数和中值处有一条线段，箱形末端延伸出去的直线称为触须，表示数据向极大和极小方向延伸的程度。触须末端以外的数据为异常值，如果在触须外没有数据，则底部触须有一点，点的颜色与触须的颜色相同。

● boxplot(X, notch)　当 notch = 1 时，生成一个带刻槽的箱形图。刻槽用图形表示箱形比较中均值不确定性的稳健性估计。默认时 notch = 0，生成一个矩形箱形图。

● boxplot(X, notch, 'sym')　其中，'sym'为一图形标记，允许你控制异常值的标注（默认时标注符号为'+'）。

● boxplot(X, notch, 'sym', vert)　当 vert = 0 时，箱形水平（默认时，vert=1，箱形垂直）。

● boxplot(X, notch, 'sym', vert, whis)　可以指定'whiskers'的长度。whis 参数用内四分位数极差（IQR）的函数来定义触须的长度。默认时，该函数为 1.5*IQR。若 whis=0，则用图形标记'sym'来显示所有箱形以外的数据值。

【例 12-1】

```
x1 = normrnd(5,1,100,1);
x2 = normrnd(6,1,100,1);
x = [x1 x2];
boxplot(x,1)
```

生成图如图 12-1 所示。

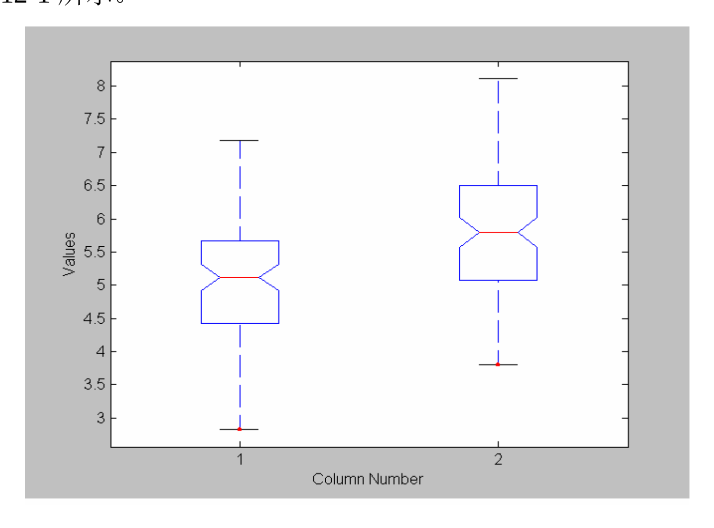


图 12-1　箱形图

12.2　经验累加分布函数图

利用 cdfplot 函数绘制经验累加分布函数图。该函数的调用格式如下：

● cdfplot(X) 对于 X 向量中的数据，生成并显示一个经验累加分布函数的图形。经验累加分布函数定义为 X 的值小于或等于 *x* 的比例。

本图有助于确定数据样本的分布。可将一个理论累加分布函数叠加到同一个图中，对样本经验分布和理论函数进行比较。

● kstest，kstest2 和 lillietest 函数对经验累加分布函数得到的统计量进行检验。将发现由 cdfplot 函数生成的经验累加分布函数对于理解源于这些函数的输出是有益的。

● H = cdfplot(X) 返回句柄到累加分布函数曲线。

● [h, stats] = cdfplot(X) 返回具有表 12-1 中域值的 stats 结构。

表 12-1　stats 结构属性的含义

域　　值	含　　义
stats.min	最小值
stats.max	最大值
stats.mean	样本均值
stats.median	样本中值
stats.std	样本标准差

【例 12-2】　生成一个正态样本和该数据的经验累加分布函数图。

```
x = normrnd(0,1,50,1);
cdfplot(x)
```

生成图如图 12-2 所示。

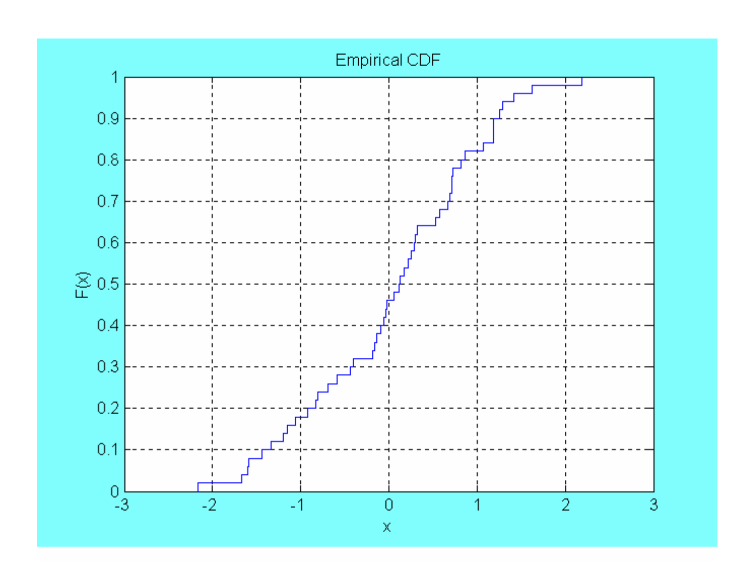


图 12-2　经验累加分布函数图

12.3　误差条图

误差条图可以直观地表示数据的离散特征。它用一条短横线表示均值，用短横线上下延伸的触须表示置信区间、均值标准误差或标准差。

用 errorbar 函数绘制沿曲线的误差条图。调用格式如下：

● errorbar(X, Y, L, U, symbol) 生成 X，Y 的误差条图。误差条图的长度由 L 和 U 来指定。X，Y，L 和 U 的长度必须相等。若 X，Y，L 和 U 为矩阵，则为每一列单独生成直线。误差条至直线的距离由 U(i)和 L(i)来确定，symbol 为一字符串，控制线型、图形标记和误差条图的颜色。

● errorbar(X, Y, L) 在 *X-Y* 坐标上绘制 Y 的对称误差条。

● errorbar(Y, L) 用误差条[Y-L　Y+L]绘制 Y。

【例 12-3】

```
lambda = (0.1:0.2:0.5);
r = poissrnd(lambda(ones(50,1),:));
[p,pci] = poissfit(r,0.001);
L = p - pci(1,:)
U = pci(2,:) - p
errorbar(1:3,p,L,U,'+')
L =
    0.1200    0.1600    0.2600
U =
    0.2000    0.2200    0.3400
```

生成图如图 12-3 所示。

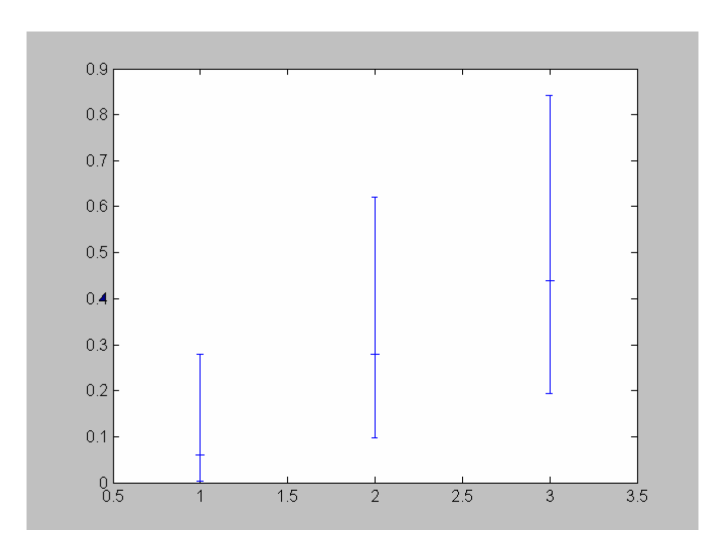


图 12-3　误差条图

12.4　函数交互等值线图

用 fsurfht 函数生成函数的交互式等值线图。调用格式如下：

● fsurfht('fun', xlims, ylims) 生成由变量 fun 指定的交互式等值线图。X 轴的限制由 xlims = [xmin xmax]来指定，Y 轴的限制由 ylims 指定。

● fsurfht('fun', xlims, ylims, p1, p2, p3, p4, p5) 允许给‘fun’函数提供 5 个选项参数。fun 的前面两个变量分别为 X 轴变量和 Y 轴变量。

图中有垂直参照线和水平参照线，两者的交点对应于当前点的 x 值和 y 值。可以通过拖拉这些带点的白色参考线来查看计算的 z 值（在图形上方）。另外，也可以通过在 X 轴和 Y 轴的文本框中输入 z 值和 z 值来得到指定的 z 值。

【例 12-4】 装载数据 gas.mat，绘出高斯似然函数的图形。

```
load gas
```

编写 M 文件 gauslike.m：

```
function z = gauslike(mu,sigma,p1)
n = length(p1);
z = ones(size(mu));
for i = 1:n
z = z .* (normpdf(p1(i),mu,sigma));
end
```

gauslike 调用 normpdf 函数，认为数据样本固定，参数 μ 和 σ 为变量。假设天然气价格为正态分布，图形显示样本的似然曲面。

```
fsurfht('gauslike',[112 118],[3 5],price1)
```

生成图 12-4。

样本均值为最大值处的 x 值，单样本的标准差不是最大值处的 y 值。

```
mumax = mean(price1)
mumax =
```

```
    115.1500
sigmamax = std(price1)*sqrt(19/20)
sigmamax =
    3.7719
```

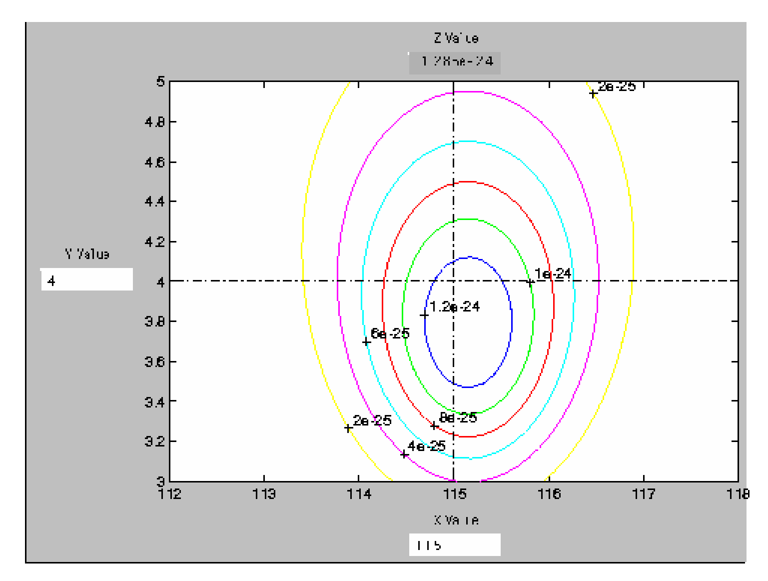


图 12-4　高斯似然函数的图形

12.5　交互画线

用 gline 函数在图上交互地画线。调用格式如下：

- glline(fig) 通过在图中用鼠标单击两个端点来绘制直线。
- h = gline(fig) 返回直线的句柄到 h 中。不带输入参数的 gline 函数在当前图形中绘直线。

【例 12-5】　利用“gline”命令打开一个空白的二维坐标系图形，然后通过单击直线按钮在图中交互地画了一个五角星。其效果如图 12-5 所示。

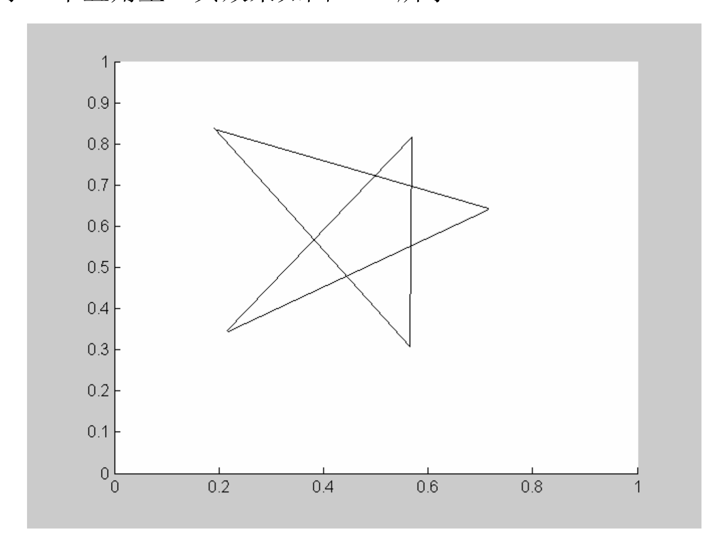


图 12-5　交互画线

12.6　交互点标注

使用 gname 函数，用案例名或案例号来标注图中的点。调用格式如下：

● gname('cases') 显示图形窗口，用鼠标确定十字线的位置，然后在希望标注的点附近单击一次，完成后，按回车键，将在你单击的每一个点附近显示标记。'cases'是一个字符串矩阵。矩阵中的每一行是数据点的案例名。

● gname 函数不带变量时，用案例号来标注每一个案例。

● h = gname(cases, line_handle) 返回图中文本对象的句柄向量。若图中有一个以上的直线对象，用 scalar, line_handle 定义正确的直线。

【例 12-6】 下面使用城市电视收视率数据集来确定哪些城市的教育和艺术是最好的，哪些是最差的。

```
load cities
education=ratings(:,6);arts=ratings(:,7);
plot(education,arts, '+')
gname(names)
```

生成图 12-6。

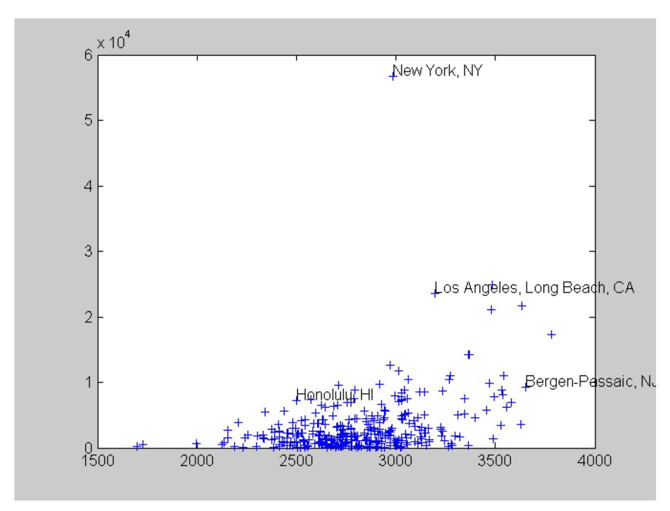


图 12-6　交互点标注

12.7　散点矩阵图

用 gplotmatrix 函数根据分组数据绘散点矩阵图。调用格式如下：

● gplotmatrix(x, y, g) 创建一个矩阵散点图。所有图形用分组变量 g 进行区分。x 和 y 是具有相同行数的矩阵。如果 x 有 *p* 列，y 有 *q* 列，则图中包含一个 $p \times q$ 的散点图矩阵。如果忽略 y 或将它指定为空矩阵，则 gplotmatrix 函数创建一个由 x 的列两两之间共同创建的平方散点图矩阵。g 可以是向量、字符串数组或字符串单元数组。g 必须与 x 和 y 具有相同的行数。*g* 值相同的点被放到同一组，在图中用相同的标记和颜色进行显示。g 也可以是一个包含几个分组变量（如{ G1　G2　G3}）的单元数组。

● gplotmatrix(x, y, g, 'clr', 'sym', siz) 给每个分组指定颜色、标记类型和大小。clr 为颜色字符串数组，可被绘图函数识别。默认时，'clr' = 'bgrcmyk'。'sym'为字符串数组，也可被绘图函数识别，其默认值为' . '。siz 为向量大小，默认值由'defaultlinemarkersize'属性确定。

● gplotmatrix(x, y, g, 'clr', 'sym', siz, 'doleg') 图例显示控制。默认时，'doleg' = 'on'，显示图例；'doleg' = 'off'，则取消显示。

● gplotmatrix(x, y, g, 'clr', 'sym', siz, 'doleg', 'dispopt') 控制 x-x 图形矩阵对角线上的显示，允许值为'none'，'hist'或'variable'，意义分别为

'none' 对角线空白；

'hist' 绘直方图(为默认选项)；

'variable' 写变量名。

● gplotmatrix(x, y, g, 'clr', 'sym', siz, 'doleg', 'dispopt', 'xnam', 'ynam') 指定 x 和 y 数组中列的名称。这些名称用于标注 X 和 Y 轴。'xnam'和'ynam'必须为字符串数组，分别用一行对应 x 和 y 的一列。

● [h, ax, bigax] = gplotmatrix(...) 返回 3 个句柄数组。h 为图中直线的句柄。ax 为单个图形的坐标轴句柄。bigax 为整个图形矩阵的坐标轴（隐藏）句柄。

【例 12-7】 装载 cities 数据。ratings 数组中有不同城市 9 个分类的比率（分类名在 categories 数组中）。group 为一代码，最大的城市的 group 值为 2。现在用前面 3 个分类与另外 4 个分类绘制散点矩阵图。

```
load discrim
gplotmatrix(ratings(:,1:3),ratings(:,4:7),group)
```

输出图形为一图形数组，如图 12-7 所示。图中每一个城市用一种不同的颜色代表。如果在图中指定颜色和标记，用分类值标注坐标轴，则该图将更易于阅读。如图 12-8 所示。

```
gplotmatrix(ratings(:,1:3),ratings(:,4:7),group,...
          'br','.o',[],'on','',categories(1:3,:),...
categories(4:7,:))
```

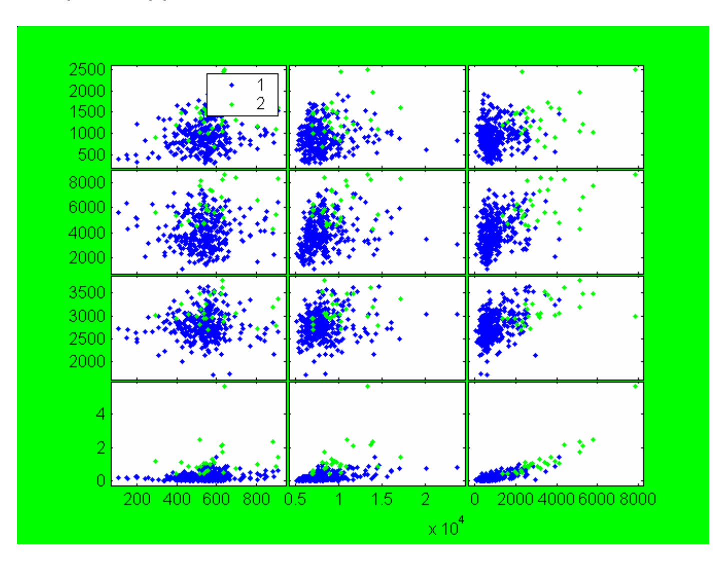


图 12-7 散点矩阵图

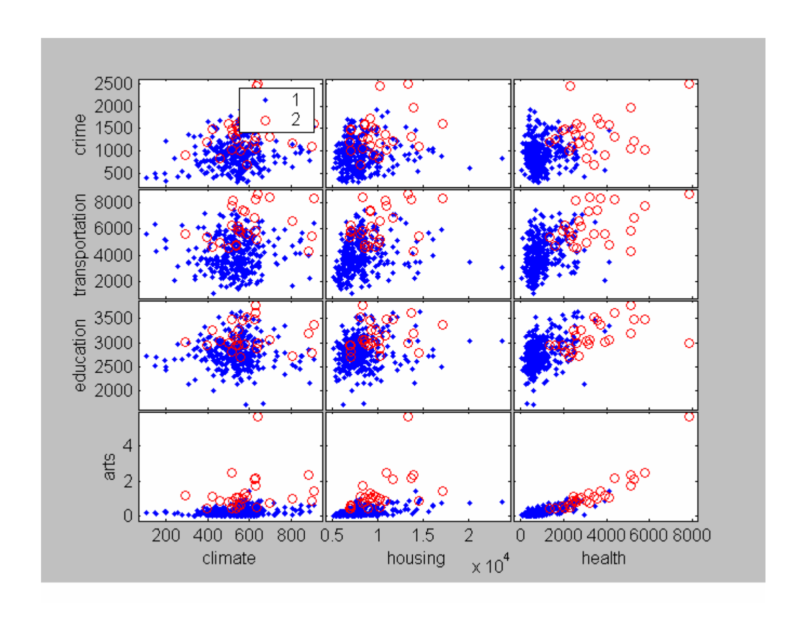


图 12-8　散点矩阵图（每个城市用一种颜色代表）

12.8　散点图

用 gscatter 函数根据分组数据绘散点图，其调用格式如下：

● gscatter(x, y, g) 创建 x 和 y 的散点图，用 g 进行分组。其中，x 和 y 为向量，具有相同的大小，g 可以是向量、字符串数组或字符串单元数组。具有相同 g 值的点分在一组，在图中用相同的标记和颜色表示。另外，g 可以是包含一些分组变量（如{G1 G2 G3}）的单元数组。

● gscatter(x, y, g, 'clr', 'sym', siz) 指定每组的颜色、标记类型和大小。默认时，'clr' = 'bgrcmyk'。'sym'为字符串数组，也可被绘图函数识别，其默认值为' . '。siz 为向量大小，默认值由'defaultlinemarkersize'属性确定。

● gscatter(x, y, g, 'clr', 'sym', siz, 'doleg') 控制是否在图中显示图例。默认时，'doleg' = 'on', 显示图例；若'doleg' = 'off'，则取消显示。

● gscatter(x, y, g, 'clr', 'sym', siz, 'doleg', 'xnam', 'ynam')指定名称，作为 X 轴和 Y 轴的标签。如果 x 和 y 的输入为简单的变量名，而且 xnam 和 ynam 被忽略，则 gscatter 函数用变量名标注坐标轴。

● h = gscatter(...) 在图中返回直线的句柄数组。

【例 12-8】 装入城市数据，调查气候（第 1 列）与房屋建造（第 2 列）之间的关系。在图中指定了进行城市区分的颜色和标记。

```
load discrim
gscatter(ratings(:,1),ratings(:,2),group,'br','xo')
```

生成的分组散点图如图 12-9 所示。

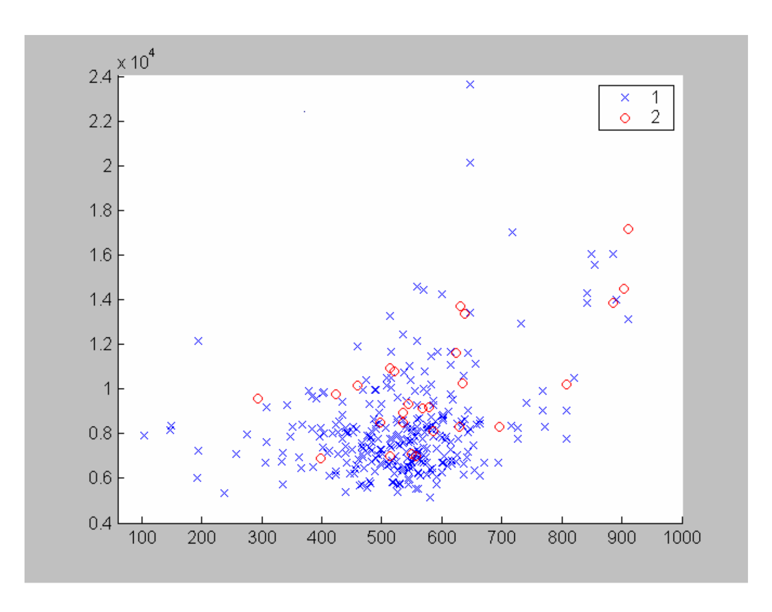


图 12-9　分组散点图

12.9　添加最小二乘拟合线

用 lsline 函数进行直线的最小二乘拟合，其调用格式如下：

- lsline 函数在当前轴中每一直线对象上添加最小二乘直线（不能用线型'-','- -'和'. -'）。
- h = lsline 返回直线对象的句柄。

【例 12-9】

```
y = [2 3.4 5.6 8 11 12.3 13.8 16 18.8 19.9]';
plot(y,'+');
lsline;
```

生成的散点的最小二乘拟合线如图 12-10 所示。

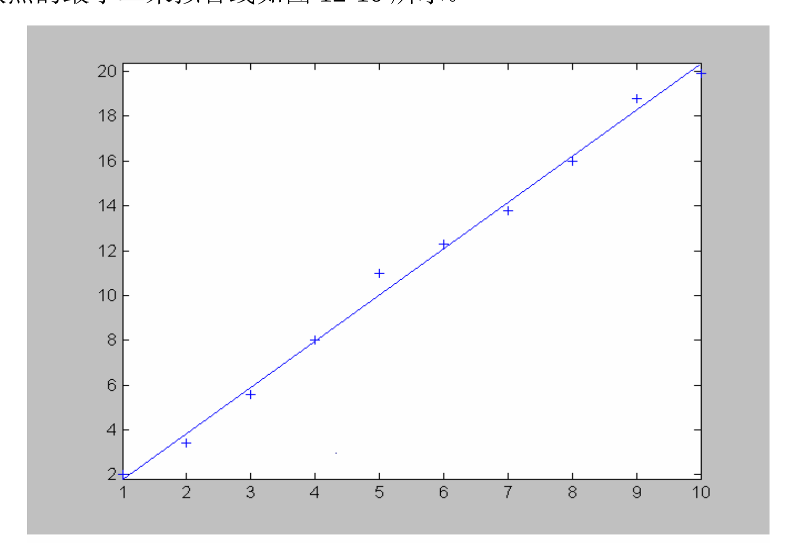


图 12-10　散点的最小二乘拟合线

12.10　正态概率图

正态概率图用于正态分布检验。数据中的每个值对应于图中的一个“+”号，其位置由点的值和经验概率共同决定，实线连接 25%和 75%百分位数，并代表稳健性线性拟合（稳健性指对样本极值不敏感）。虚线从实线两侧延伸到样本末端。正态概率图中 Y 轴的比例是不均匀的。它表示概率，值界于 0 到 1 之间。Y 轴上的短线标记之间的距离与正态分布数量之间的距离相匹配，在中值附近数值比较集中，在远离中值的方向上对称地伸展开去。如果所有的数据点都落在直线附近，则认为数据服从正态分布，若不服从，则“+”连成一条曲线。

使用 normplot 函数，用正态概率图进行正态性检验，其调用格式为：

● normplot(X) 显示数据 X 的正态概率图，若 X 为矩阵，则为 X 的每一列生成一条直线。该图中的样本数据用图形标记‘+’显示。在图中添加 X 中每一列数据四分之一和四分之三处的连线。该线可以看做样本次序统计量的稳健性直线拟合。它可以帮助评价数据的线性特征。若数据源于正态分布，则图形呈直线形，否则为曲线。

● h = normplot(X) 返回图中直线的句柄。

【例 12-10】 生成一个正态样本，绘数据的正态概率图。

```
x = normrnd(0,1,50,1);
h = normplot(x);
```

生成的正态分布概率图如图 12-11 所示。该图呈线性，表示可以认为样本服从正态分布。

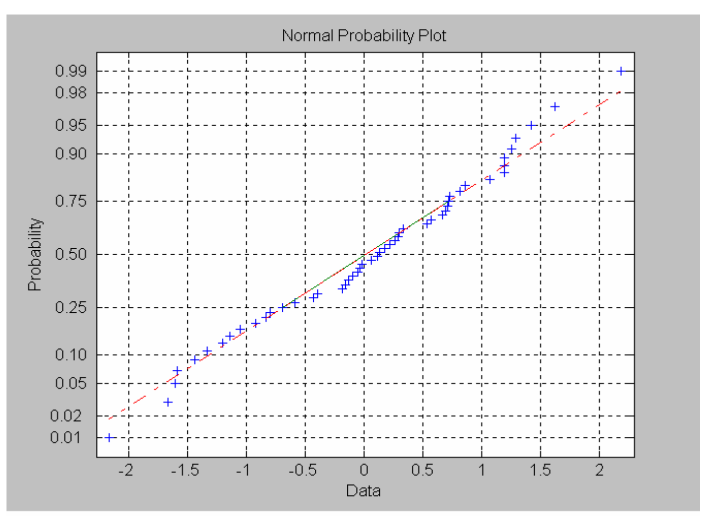


图 12-11　正态分布概率图

12.11　帕累托图

帕累托图在形式上是条形图和线形图的组合。不同的是，帕累托图中的条形图单元根据条形对应量的主次关系或重要性程度从前到后进行了重新排序，并且在条形图上方显示对应于条形图单元的百分比累加曲线。因此，从帕累托图中可以一目了然地看出各组成成分或影响因素中的主次关系，这对于在管理活动或科学分析中发现主要问题并集中精力解决主要问题是十分有用的。

使用 pareto 函数生成帕累托图，其调用格式如下：

● pareto(y,names) 向量 y 中的数值以降序绘成条形，每一个条形用字符串矩阵（names）中的相关值来标注。用 y 中的对应元素的系数来标注每一个条形。条形上面的直线显示累加百分比。

● pareto(y,'names') 用'names'矩阵的行来标注每一个条形，对应于 y 的绘图单元。

● h = pareto(...) 返回填充部分与直线句柄的组合。

【例 12-11】 根据有不同类型缺陷的产品个数数据绘制帕累托图。

```
defects = ['pits ';'cracks';'holes ';'dents '];
quantity = [5 3 19 25];
pareto(quantity,defects)
```

生成的帕累托图如图 12-12 所示。

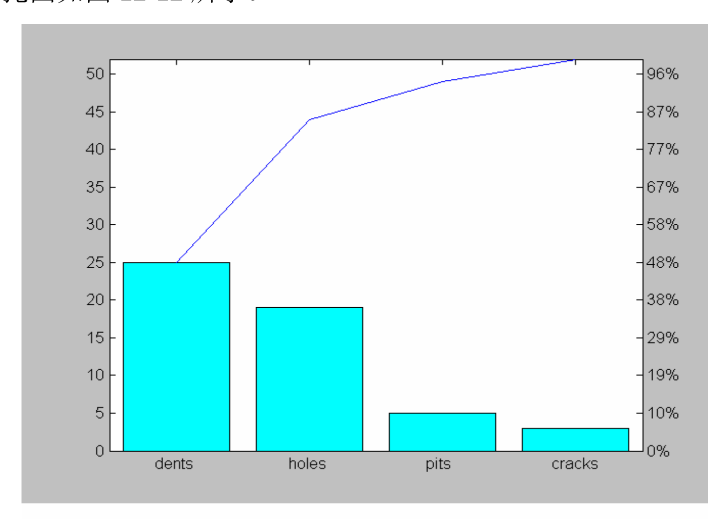


图 12-12 帕累托图

12.12 q-q 图

q-q 图用变量数据分布的分位数与所指定分布的分位数之间的关系曲线来检验数据的分布。如果两个样本来自同一分布，则图中数据点呈现直线关系，否则为曲线关系。该图中将样本数据用图形标记‘+’显示。在图中将每一个分布的四分之一和四分之三处进行连线（可以看成是两个样本次序统计量的稳健线性拟合）。该线可用来评价数据的线性特征。

用 qqplot 函数生成两个样本的 q-q 图，其调用格式如下：

● qqplot(X) 显示 X 的样本值与服从正态分布的理论数量值之间的 q-q 图。如果 X 的分布为正态分布，则图形接近直线。

● qqplot(X,Y) 显示两个样本的 q-q 图。若样本不是来自于相同的分布，则图形将是线性的。对于矩阵 X 和 Y，q-q 图为每一个配对列显示分隔线。

● h = qqplot(X, Y, pvec) 返回直线的句柄到 h 中。

【例 12-12】 生成具有不同均值和标准差的两个正态样本。然后作两个样本的 q-q 图。

```
x = normrnd(0,1,100,1);
y = normrnd(0.5,2,50,1);
```

```
qqplot(x,y);
```

生成图 12-13。

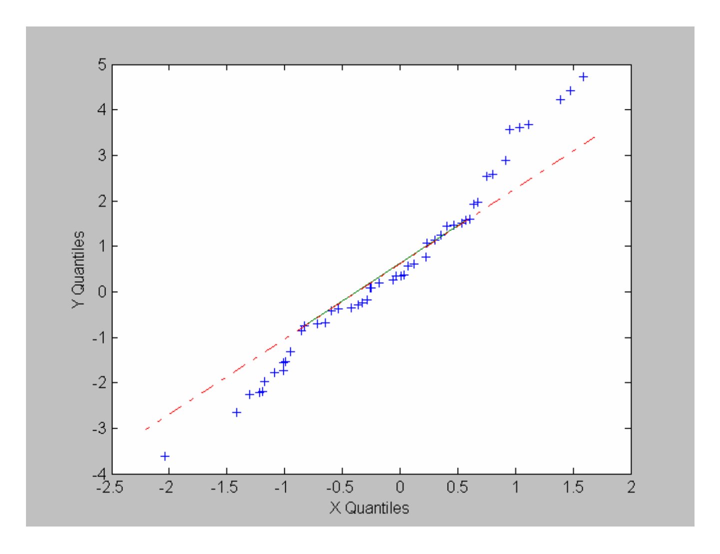


图 12-13 两个样本来自同一分布的 q-q 图

可见，虽然两个样本的参数和大小不同，图中的直线关系仍然显示两个样本来自同一分布。

下面的例子显示了两个样本不是来自同一分布的情况（生成的 q-q 图如 12-14 所示）：

```
x=normrnd(5,1,100,1);
y=weibrnd(2,0.5,100,1);
qqplot(x,y);
```

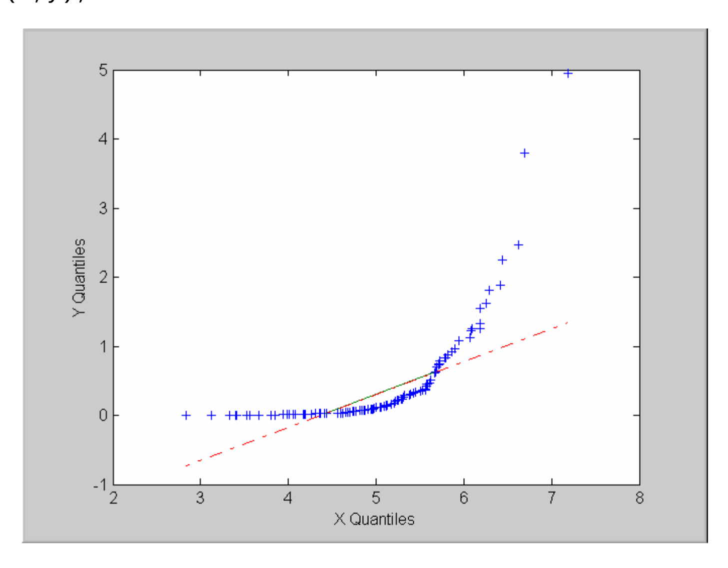


图 12-14 样本不同分布时的 q-q 图

用 q-q 图的直线关系来确保两个样本来自相同分布是不正确的。但是，从满足样本服从同分布假设的角度讲，q-q 图中的直线关系所提供的证据应该是足够了。

12.13 回归个案次序图

用 rcoplot 函数生成残差案例序号图，其调用格式如下：

● rcoplot(r, rint) 用误差条图表示回归残差的置信区间，以个案序号出现。r 和 rint 是 regress 函数的输出。

【例 12-13】

```
X = [ones(10,1) (1:10)'];
y = X * [10;1] + normrnd(0,0.1,10,1);
[b,bint,r,rint] = regress(y,X,0.05);
rcoplot(r,rint);
```

如图 12-15 所示，图形显示了残差 95%置信区间对应的误差条图。所有的误差条图通过 0 线，表示数据中没有异常值。

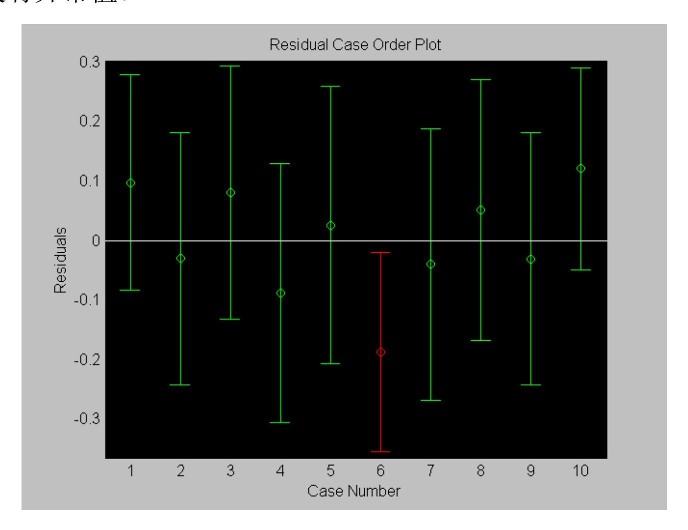


图 12-15 残差个案次序图

12.14 参考多项式曲线

用 refcurve 函数在当前图中添加多项式曲线，其调用格式如下：

● refcurve 给当前轴添加多项式 p 的图形。n 次多项式可以写成：

$$y = p_1 x^n + p_2 x^{(n-1)} + \ldots + p_n x + p_{n+1}$$

● h = refcurve(p) 返回曲线的句柄。

【例 12-14】 根据火箭飞行高度和时间数据绘图，并添加参考曲线显示火箭的理论高度（假设没有空气摩擦）。火箭的初速度为 100m/s。

```
h = [85 162 230 289 339 381 413 437 452 458 456 440 400 356];
plot(h,'+')
refcurve([-4.9 100 0])
```

生成图 12-16。

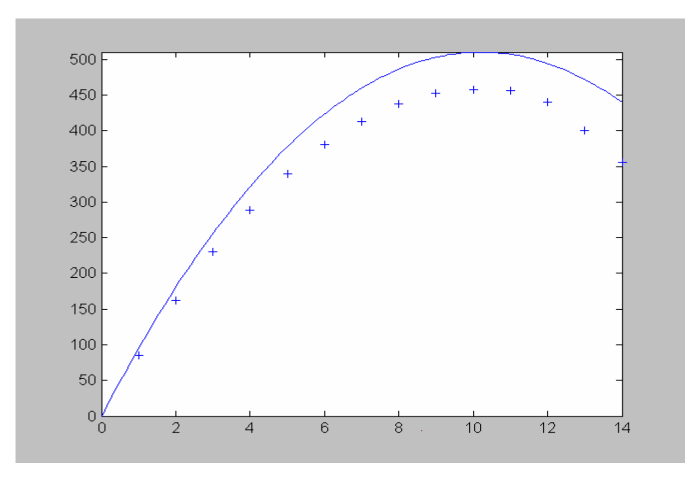


图 12-16　多项式拟合线

12.15　添加参考线

用 refline 函数在当前轴上添加参考线，其调用格式如下：

- refline(slope, intercept)根据给定的斜率和截距在当前轴上添加参考线。
- refline(slope) 其中 slope 为一二元素向量，在图中添加下面的直线：

  y = SLOPE(2) + SLOPE(1)x
- h = refline(slope, intercept) 返回直线的句柄。

不带变量的 refline 函数在当前图中的每一个直线对象上添加最小二乘拟合线（除去线型 '-' , '- -' 和 '. -'）。该操作的效果同 lsline 函数。

【例 12-15】

```
y = [3.2 2.6 3.1 3.4 2.4 2.9 3.0 3.3 3.2 2.1 2.6]';
plot(y,'+')
refline(0,3)
```

其结果如图 12-17 所示。

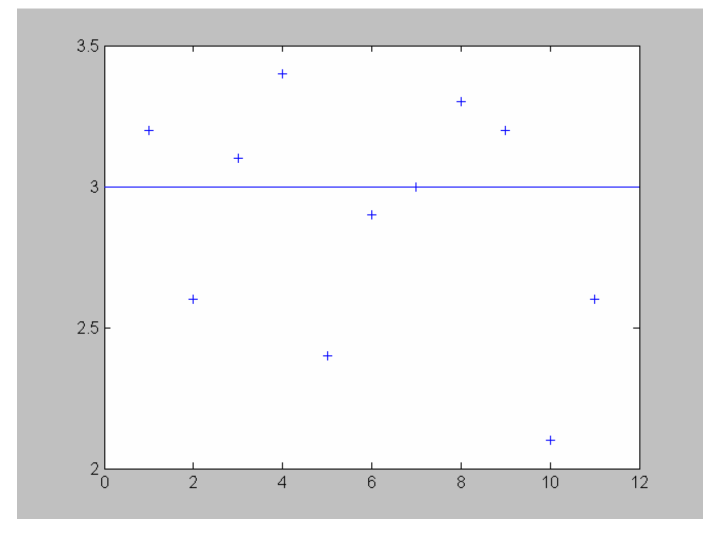


图 12-17　添加参考线

12.16　交互插值等值线图

用 surfht 函数交互地绘制等值线图，其调用格式如下：

● surfht(Z) 将 Z 矩阵视为平面上的高度，交互地绘制等值线图。x 值为 Z 的列指数，y 值为 Z 的行指数。

● surfht(x, y, Z)中 x 和 y 为向量，指定等值线图上的 X 轴和 Y 轴。x 的长度必须与 Z 的列数匹配，y 的长度必须与 Z 的行数匹配。

图 12-18（例 12-16 生成的图）中有垂向和水平向的参考线，它们的交点定义当前点的 x 值和 y 值。可以同时拖曳这些白色的点线，并观察图形上面的 z 值。另外，还可以分别在 X 轴和 Y 轴的编辑文本域中输入 x 值和 y 值，得到指定的内插 z 值。

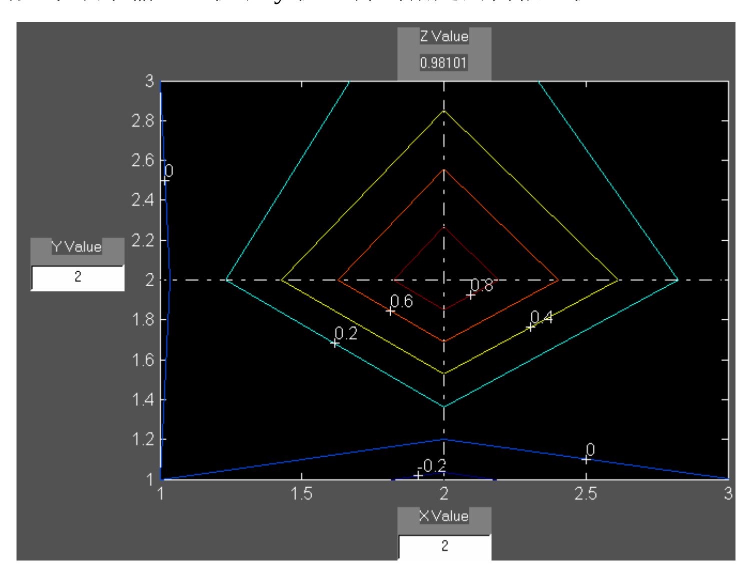


图 12-18　交互插值等值线图

【例 12-16】

```
Z=peaks(3);
surfht(z)
```

12.17　威布尔图

威布尔图可以判断数据是否服从威布尔分布。许多可靠性分析，假设寿命服从威布尔分布，用该图可以进行检验。

用 weibplot 函数生成威布尔概率图，其调用格式如下：

● weibplot(X) 显示 X 中数据的威布尔概率图。若 X 为矩阵，weibplot 函数显示每一列数据的图形。

● h = weibplot(X)返回图中直线的句柄。

绘制威布尔概率图的目的是用图形显示 X 中的数据是否来自威布尔分布。若数据服从威布尔分布，则图形应该是线性的。若数据服从其他分布，则图形是曲线形的。

【例 12-17】　下面的语句生成的威布尔图，如图 12-19 所示。

```
r = weibrnd(1.2,1.5,50,1);
weibplot(r)
```

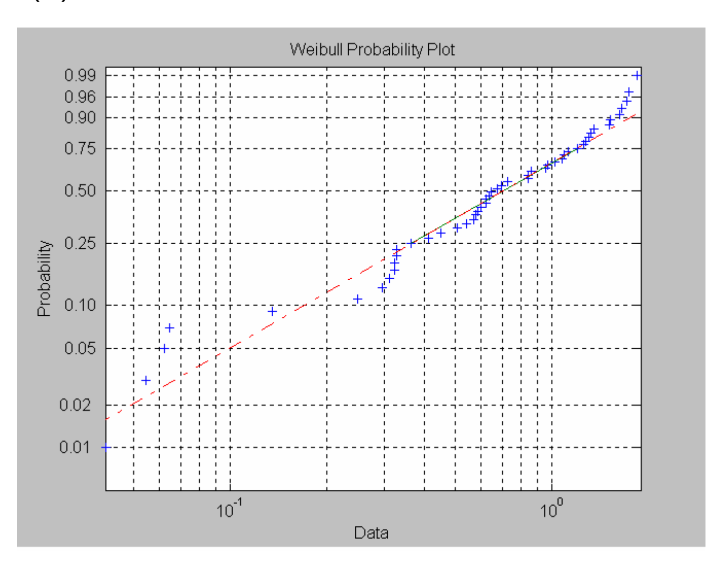


图 12-19　威布尔图

第 13 章　文件输入/输出

MATLAB 统计工具箱提供了几个函数，利用它们，可以实现文件的输入和输出。

13.1　文件输入

利用 tblread 函数和 caseread 函数，可以实现从文件中读取数据。

1．tblread 函数

利用该函数从文件系统中读入表格间隔的固定格式文本数据，其调用格式如下：

● [data, varnames, casenames] = tblread　显示 File Open 对话框，选择需要输入的数据文件。文件的格式要求：第 1 行为变量名，第 1 列为案例名，数据从(2,2)的位置开始读入。

● [data,varnames, casenames] = tblread(filename)　允许在当前路径中输入指定的文件名，也可以输入完整路径的任何文件的文件名。

● [data, varnames, casenames] = tblread(filename, 'delimiter')　允许在文件中指定数据之间的分隔符。合法的分隔符包括 'tab' (表格间隔)、'space' (空格间隔)或 'comma' (逗号间隔)。

tblread 函数返回 3 个值：

- data　为数值矩阵，矩阵中的每一个元素对应于每一个变量-案例匹配对；
- varnames　在第 1 行中包含变量名的字符串矩阵；
- casenames　在第 1 列中包含每一个案例名的字符串矩阵。

【例 13-1】　MATLAB 统计工具箱目录下有一个数据文件 sat.dat，现在读取它的数据。

```
[data,varnames,casenames] = tblread('sat.dat')
data =
   470   530
   520   480
varnames =
Male
Female
casenames =
Verbal
Quantitative
```

2．caseread 函数

利用该函数从文件中阅读案例名，其调用格式如下：

● names = caseread(filename)　读文件名的内容并返回字符串矩阵 names。filename 是当前路径上的文件名或给定完整路径的任何文件名。caseread 将每一行作为一个单独的案例。

● names = caseread　显示 File Open 对话框，以交互地选择输入文件。

【例 13-2】　本例使用由 casewrite 函数（参见下一节内容）创建的文件 months.dat。

```
type months.dat
January
```

```
February
March
April
May
names = caseread('months.dat')
names =
January
February
March
April
May
```

13.2　文件输出

利用 tblwrite 函数和 casewrite 函数实现数据输出。

1．tblwrite 函数

利用该函数将表格间隔的数据输出到文件系统，其调用格式如下：

- tblwrite(data, 'varnames', 'casenames') 显示 File Open 对话框，用于交互地指定表格间隔数据输出文件。文件的格式要求：第 1 行为变量名，第 1 列为案例名，数据从(2,2)的位置开始读入。'varnames' 为一包含变量名的字符串矩阵；'casenames' 为一包含第 1 列中每个案例名的字符串矩阵；data 为一数值矩阵，矩阵中的每一个元素对应于每一个变量-案例匹配数据。
- tblwrite(data, 'varnames', 'casenames', 'filename') 允许用命令行的形式输入当前路径的文件名 'filename' 或给定 'filename' 的完整路径名。

【例 13-3】 继续使用 tblread 函数的例子：

```
tblwrite(data,varnames,casenames,'sattest.dat')
type sattest.dat
               Male   Female
Verbal         470     530
Quantitative    520     480
```

2．casewrite 函数

利用该函数将案例名从字符串矩阵写到一个文件中，其调用格式如下：

- casewrite (strmat, filename) 将 strmat 的内容写到 filename 文件中。strmat 的的每一行代表一个案例名。filename 为当前路径上的文件名或给定完整路径时的任意文件。
- casewrite 将每一个名称写到 filename 文件的对应单行上。
- casewrite(strmat) 显示 File Open 对话框，以交互地指定输出文件。

【例 13-4】 下面利用 casewrite 函数创建一个新文件 months.dat。

```
strmat = str2mat('January','February','March','April','May')
strmat =
January
February
March
April
```

```
May
casewrite(strmat,'months.dat')
type months.dat
January
February
March
April
May
```

代码首先用 strmat 函数把 5 个表示月份的英文字母组合成一个字符距阵,然后用 casewrite 函数把该矩阵写到文件 months.dat 中。注意，在字符矩阵中，所有字符串的长度是相等的，以最长的字符串为标准，较短的字符串要在末尾用空格补齐。

第 14 章　统 计 演 示

14.1　交互式方差分析工具

用 aoctool 函数绘制进行方差模型拟合和预测分析的交互图。调用格式如下：

● aoctool(x, y, g) 对于每个由 g 数组的值定义的分组，用单条直线拟合 x 向量和 y 向量的列数据。这些类型的模型包括单因素多元方差分析等。输出由 3 个图形组成：数据和预测曲线的交互图，一个单因素方差分析表和一个参数估计表。可以使用图形改变模型，检验模型的不同部分。

● aoctool(x, y, g, alpha) 确定预测区间的置信水平。置信水平为 100*(1-alpha)%。alpha 的默认值为 0.05。

● aoctool(x, y, g, alpha, xname, yname, gname) 指定 x、y 和图表中 g 变量的名称。如果为 x、y 和 g 变量输入简单的变量名，则 aoctool 函数将使用这些名称。如果要为其中的某个变量指定表达式，则可以通过在表达式处提供这些变量来指定名称。例如，如果作为 x 变量输入 m(:,2)，可以选择输入'Col 2'，并作为 xname 变量。

● aoctool(x, y, g, alpha, xname, yname, gname, 'displayopt') 当'displayopt'设置为'on'（默认设置）时，激活图表显示；当'displayopt'设置为'off'时，则取消显示。

● aoctool(x, y, g, alpha, xname, yname, gname, 'displayopt', 'model') 指定进行拟合的初始模型。'model'的值可以是下面的任意一个：

'same mean'——拟合单个均值，忽略分组；
'separate means'——分别拟合每个分组的均值；
'same line'——拟合单条直线，忽略分组；
'parallel lines'——用单条直线拟合每个分组，但包含平行线；
'separate lines'——用单条直线拟合每个分组，没有约束。

● h = aoctool(...) 返回图中直线对象的句柄向量。

● [h, atab, ctab] = aoctool(...) 返回包含方差分析表（atab）中入口的单元数组和系数估计表(ctab)。

● [h, atab, ctab, stats] = aoctool(...) 返回 stats 结构，可用于进行多元比较检验。ANOVA 表输出包含斜率和截距均相同的假设检验。有时进行检验，确定哪些数据对具有显著差异，哪些没有是很合适的。可以将 stats 结构作为输入，利用 multcompare 函数进行该检验。可以检验斜率、交互作用或总体边缘均值。

【例 14-1】 本例演示如何无交互地进行不同模型的拟合。首先装入更小的汽车数据集并拟合单斜率模型，然后进行系数估计。

```
load carsmall
[h,a,c,s] = aoctool(Weight,MPG,Model_Year,0.05,...
                    '','','','off','separate lines');
c(:,1:2)
```

```
ans =
    'Term'          'Estimate'
    'Intercept'     [45.97983716833132]
    '  70'          [-8.58050531454973]
    '  76'          [-3.89017396094922]
    '  82'          [12.47067927549897]
    'Slope'         [-0.00780212907455]
    '  70'          [ 0.00195840368824]
    '  76'          [ 0.00113831038418]
    '  82'          [-0.00309671407243]
```

概略地讲，MPG 与 Weight 之间的相关曲线在靠近 45.98 的地方有交互效应，在靠近 -0.0078 的地方有一个斜率。每个组的系数与这些值之间都有不同程度的偏离。例如，1970 年产的汽车的交互效应为 45.98-8.58 = 37.40。

下一步，我们试图用平行线进行拟合。

```
[h,a,c,s] = aoctool(Weight,MPG,Model_Year,0.05,...
                   '','','','off','parallel lines');
c(:,1:2)
ans =
    'Term'          'Estimate'
    'Intercept'     [43.38984085130596]
    '  70'          [-3.27948192983761]
    '  76'          [-1.35036234809006]
    '  82'          [ 4.62984427792768]
    'Slope'         [-0.00664751826198]
```

这里，对于每个组又有了单独的交互效应，但这时要求斜率是相同的。

14.2　交互式经验分布函数工具

利用 disttool 命令生成多种概率分布的交互式函数图形（或概率密度图）。

disttool 命令打开一个图形用户界面，观察在 cdf 图和 pdf 图上改变参数带来的影响。在图中单击并拖拉垂线可以交互地在整个内域上进行评价。

通过在 X 轴编辑框中输入值或在图中拖拉垂向参考线可以评价绘图函数。对于 cdf，可以通过在 Y 轴编辑框中输入值或在图中拖拉水平参考线来评价逆函数。当鼠标穿过垂线或水平线时，鼠标形状由箭头变为十字形，表示该参考线是可以拖拉的。

在图形左上方的函数弹出式菜单中进行选择，可以改变分布函数的类型。在图形右上方的弹出式菜单中进行选择，可以在 cdfs 和 pdfs 之间切换。

通过移动滚动条或在参数名下的编辑框中输入数值，可以改变参数设置。在 paremeter 滚动条的顶部或底部编辑框中输入数值，可以改变参数的限制。

完成以上操作以后，单击"close"按钮，关闭界面。

输入 disttool 命令以后，打开交互式经验分布函数工具，如图 14-1 所示。

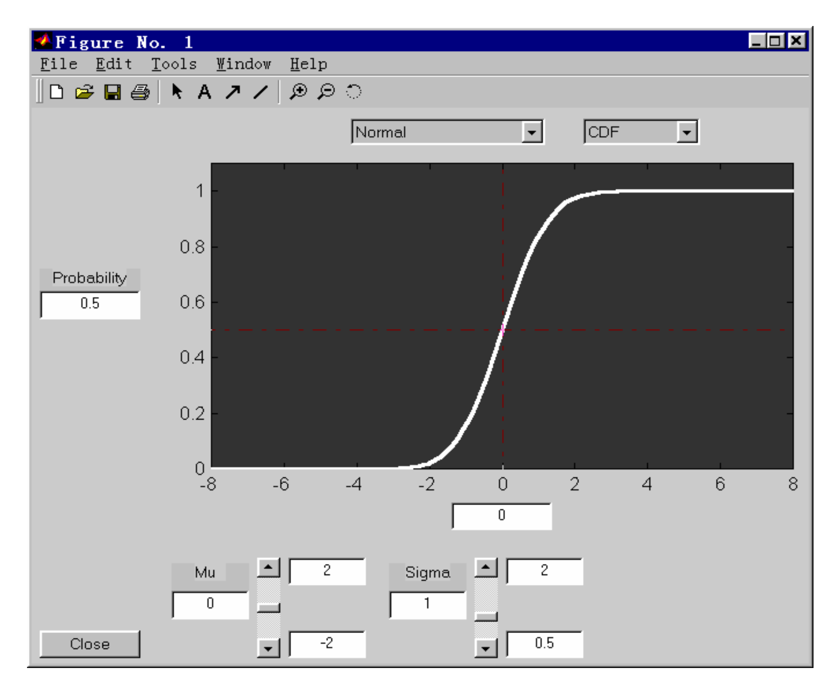


图 14-1　交互式经验分布函数工具

14.3　一般线性模型演示

用 glmdemo 函数进行一般线性模型的演示，其调用格式如下：

● glmdemo 对一般线性模型进行演示。它演示怎样用 glmfit 函数拟合一般线性模型，并怎样用 glmval 函数进行预测。

14.4　稳健回归与最小二乘拟合比较工具

用 robustdemo 函数进行稳健回归演示，其调用格式如下：

● robustdemo 对于给定的样本数据集进行稳健回归和一般最小二乘回归的演示。该函数创建一个包含样本数据 X 向量和 Y 向量的散点图，以及用最小二乘法和稳健回归法得到的两条拟合线。图形下方显示了直线方程和每个拟合的估计误差的标准离差。如果使用左边的鼠标按钮选择一个点并将它移动到一个新的位置，则两个拟合线都会更新。如果按住鼠标右键拖拉到任意点，则该点将标注其最小二乘拟合的中心化杠杆值，以及稳健拟合点的权重。

● robustdemo(X, Y) 利用指定的 X 和 Y 进行同样的演示。

【例 14-2】

```
robustdemo
```

结果如图 14-2 所示。

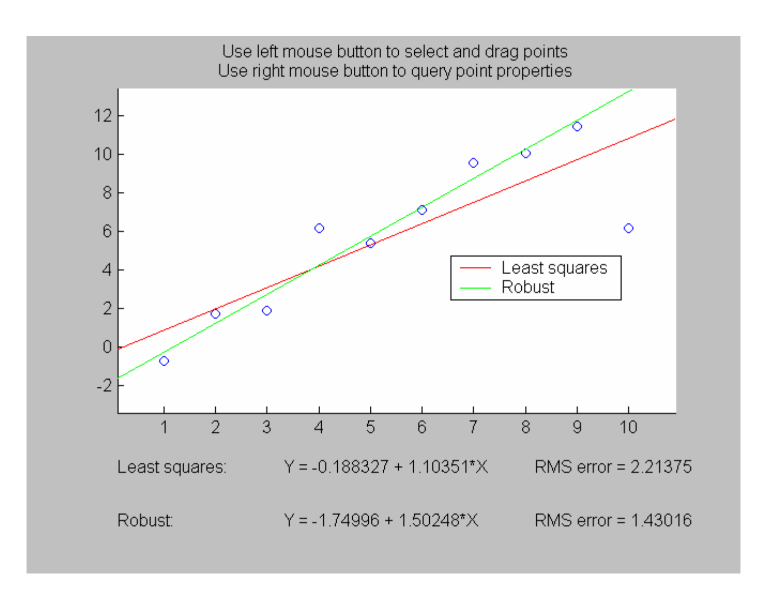


图 14-2 稳健回归与最小二乘拟合的比较

14.5 多项式拟合工具

用 polytool 函数绘制拟合多项式预测的交互图，其调用格式如下：

● polytool(x, y) 对列向量 x 和 y 的数据进行拟合，并显示结果的交互图。该图为图形用户界面，可以观察拟合多项式最高阶次改变带来的影响。该图显示拟合曲线和曲线的一个新预测值的 95%预测区间。在 Y 轴的左侧显示 y 的当前预测值和它的不确定性文本。

● polytool(x, y, n) 拟合一个最高阶次为 n 的多项式。

● polytool(x, y, n, alpha) 绘制预测值的 100(1-alpha)% 置信区间。

polytool 函数使用下面的回归模型进行最小二乘拟合。

$$y_i = \beta_0 + \beta_1 \mathrm{x}_i + \beta_2 \mathrm{x}_i^{22} + \ldots + \beta_n \mathrm{x}_i^n + \varepsilon_i$$

$$\varepsilon_i \sim N(0, \sigma^2) \quad \forall i$$

$$\mathrm{Cov}(\varepsilon_i, \varepsilon_j) = 0 \quad \forall i, j$$

在在图 14-3 所示的 X 轴编辑框中输入数值或在图中拖拉垂向参考线，可以评价函数。当鼠标越过垂线时其形状由箭头改变为十字形，表示该直线是可以拖拉的。当拖拉参考线时，*y* 的预测值将会更新。变量 *n* 控制拟合多项式的阶次。在图形顶部的弹出式菜单中进行选择，可以改变多项式的阶次。

完成以上操作以后，单击“close”按钮。

【例 14-3】

```
y=[85 162 230 289 339 381 413 437 452 458 456 440 400 356];
x=[1 2 3 4 5 6 7 8 9 10 11 12 13 14];
polytool(x,y)
```

默认时输出线性拟合图，如图 14-3 所示。在“Export”下拉式列表框中进行选择，可以输出相应的统计量。

在“Degree”窗口中输入 2，进行二次多项式拟合，得图 14-4。

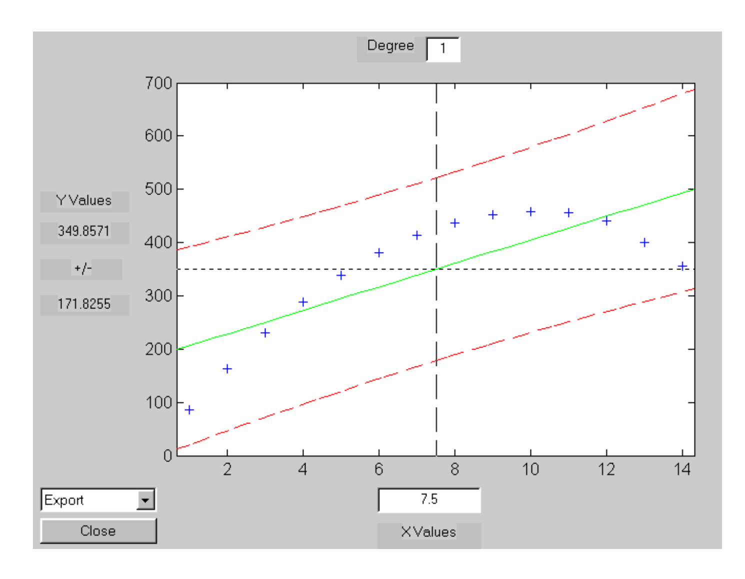


图 14-3　线性拟合

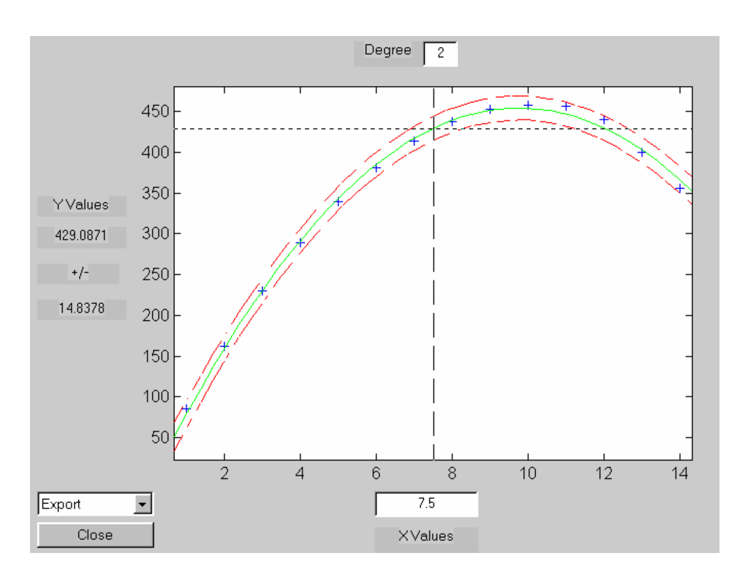


图 14-4　二次多项式拟合

14.6　随机数生成工具

使用 randtool 函数，用直方图显示如何交互地生成随机数。

randtool 命令打开一个图形用户界面，以观察在服从一定概率分布的随机样本直方图上改变参数和样本大小带来的影响。

单击“Output”按钮，输出随机数的当前设置。结果保存在变量 ans 中。另外，命令 r = randtool('output') 将随机数样本放到 ***r*** 向量中。

单击“Resample”按钮，从同一分布的总体中进行重复取样。

在图形上方的函数弹出式菜单中进行选择，可以改变分布函数。

移动滚动条或在参数名下方的编辑框中输入数值，可以改变参数的设置。在 parameter

滚动条的顶部或底部编辑框中输入数值，可以改变参数的限制。

在“Samples”编辑框中输入数值，可以改变样本的大小。

完成上面的操作以后，单击“close”按钮，关闭界面。

【例 14-4】 在命令窗口输入 randtool 命令，打开随机数生成界面，如图 14-5 所示。在“Normal”下拉式列表框中进行选择，确定生成什么分布的随机数。在“Samples”窗口中输入样本的大小。在图形下方输入对应分布的参数及其置信区间。单击“Output”按钮，生成随机数并用直方图表示。图 14-5 为服从标准正态分布的随机数的形式。

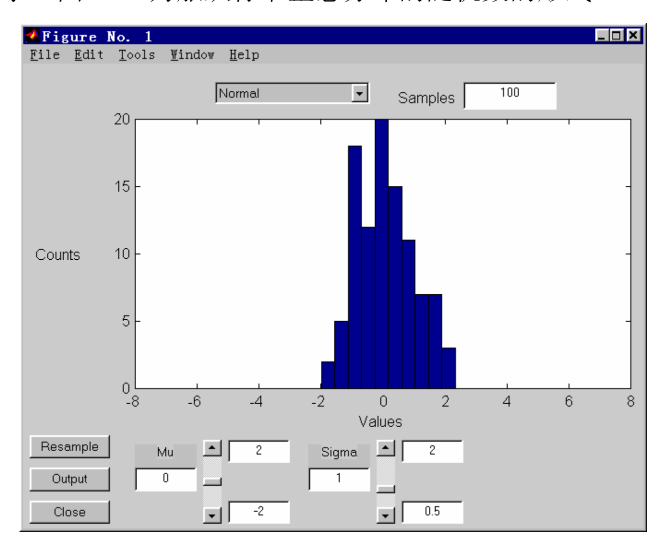


图 14-5　随机数生成工具界面

在“Normal”下拉式列表框中选择“Uniform”选项，将生成服从均匀分布的随机数，如图 14-6 所示。

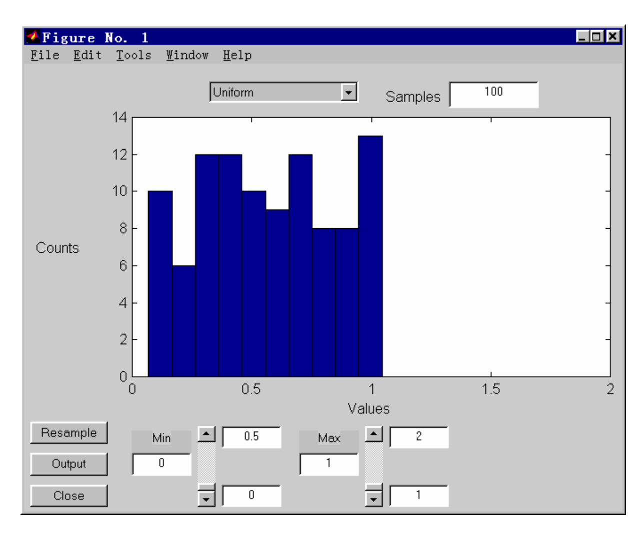


图 14-6　生成服从均匀分布的随机数

第二篇　优化工具箱

第 15 章　优化工具箱概述

在生活和工作中，人们对于同一个问题往往会提出多个解决方案，并通过各方面的论证从中提取最佳方案。最优化方法就是专门研究如何从多个方案中科学合理地提取出最佳方案的科学。由于优化问题无所不在，目前最优化方法的应用和研究已经深入到了生产和科研的各个领域，如土木工程、机械工程、化学工程、运输调度、生产控制、经济规划、经济管理等，并取得了显著的经济效益和社会效益。

用最优化方法解决最优化问题的技术称为最优化技术，它包含两个方面的内容：

（1）建立数学模型，即用数学语言来描述最优化问题。模型中的数学关系式反映了最优化问题所要达到的目标和各种约束条件。

（2）数学求解。数学模型建好以后，选择合理的最优化方法进行求解。

最优化方法的发展很快，现在已经包含有多个分支，如线性规划、整数规划、非线性规划、动态规划、多目标规划等。

利用 MATLAB 的优化工具箱，可以求解线性规划、非线性规划和多目标规划问题。具体而言，包括线性、非线性最小化，最大最小化，二次规划，半无限问题，线性、非线性方程（组）的求解，线性、非线性的最小二乘问题。另外，该工具箱还提供了线性、非线性最小化，方程求解，曲线拟合，二次规划等问题中大型课题的求解方法，为优化方法在工程中的实际应用提供了更方便、快捷的途径。

15.1　优化工具箱中的函数

优化工具箱中的函数包括下面几类，即最小化函数（见表 15-1）、方程求解函数（见表 15-2）、最小二乘（曲线拟合）函数（见表 15-3）、实用函数（见表 15-4）、大型方法的演示函数（见表 15-5）、中型方法的演示函数（见表 15-6）。

表 15-1　最小化函数表

函　　数	描　　述	函　　数	描　　述
fgoalattain	多目标达到问题	fminsearch, fminunc	无约束非线性最小化
fminbnd	有边界的标量非线性最小化	fseminf	半无限问题
fmincon	有约束的非线性最小化	linprog	线性课题
fminimax	最大最小化	quadprog	二次课题

表 15-2　方程求解函数表

函　　数	描　　述
\	线性方程求解
fsolve	非线性方程求解
fzero	标量非线性方程求解

表 15-3　最小二乘函数表

函　　数	描　　述
\	线性最小二乘
lsqlin	有约束线性最小二乘
lsqcurvefit	非线性曲线拟合
lsqnonlin	非线性最小二乘
lsqnonneg	非负线性最小二乘

表 15-4　实用函数表

函　　数	描　　述
optimset	设置参数
optimget	获取参数

表 15-5　大型方法的演示函数表

函　　数	描　　述
circustent	马戏团帐篷问题—二次课题
molecule	用无约束非线性最小化进行分子组成求解
optdeblur	用有边界线性最小二乘法进行图形处理

表 15-6　中型方法的演示函数表

函　　数	描　　述
bandemo	香蕉函数的最小化
dfildemo	过滤器设计的有限精度
goaldemo	目标达到举例
optdemo	演示过程菜单
tutdemo	教程演示

15.2　优化函数的变量

下面的 3 个表描述工具箱中优化函数的变量：表 15-7 描述输入变量，表 15-8 描述输出变量，表 15-9 描述优化选项参数结构 options。

表 15-7　输入变量表

变　　量	描　　述	调 用 函 数
A, b	***A*** 矩阵和 ***b*** 向量分别为线性不等式约束的系数和对应的右端项	fgoalattain, fmincon, fminimax, fseminf, linprog, lsqlin, quadprog
Aeq, beq	***Aeq*** 矩阵和 ***beq*** 向量分别为线性方程约束的系数和对应的右端项	fgoalattain, fmincon, fminimax, fseminf, linprog, lsqlin, quadprog
C, d	矩阵 ***C*** 和向量 ***d*** 分别为超定或未定线性系统方程组的系数和进行求解的右端项	lsqlin, lsqnonneg
f	线性方程 ***f*** '****x***=0 或二次方程 ***x***'****H*******x***+***f*** '****x***=0 中线性项的系数向量	linprog, quadprog
fun	进行优化的函数。fun 必须为行命令对象，或 M 文件、嵌入函数、或 MEX 文件的名称	fgoalattain, fminbnd, fmincon, fminimax, fminsearch, fminunc, fseminf, fsolve, fzero, lsqcurvefit, lsqnonlin
goal	目标试图达到的值向量。该向量的长度等于目标个数	Fgoalattain
H	二次方程 ***x***'****H*******x***+***f*** '****x***=0 中二次项的系数矩阵	Quadprog

续表

变　　量	描　　述	调 用 函 数
lb, ub	下限和上限向量（或矩阵）。该变量一般与 ***x*** 具有相同的大小。如果 ***lb*** 比 ***x*** 的元素少，例如如果只有 m 个，则只有 ***x*** 中的前 m 个元素给出下限，***ub*** 中的上限值服从同样的规则。可以用 -Inf（对于下界）或 Inf（对于上界）指定超边界向量。例如，若 lb(i) = -Inf，则变量 x(i)超出下界	fgoalattain, fmincon, fminimax, fseminf, linprog, lsqcurvefit, lsqlin, lsqnonlin, quadprog
nonlcon	该函数计算非线性不等式和等式。nonlcon 函数为 M 文件名或 MEX 文件名	fgoalattain, fmincon, fminimax
ntheta	半无限约束的个数	Fseminf
options	为优化选项参数结构，定义用于优化函数的参数定义	All functions
P1, P2,...	传给 fun 函数、nonlcon 变量（如果存在）或 seminfcon 变量（如果存在）的其他变量。当优化函数调用函数 fun, nonlcon 或 seminfcon 时，调用格式为 f = feval(fun,x,P1,P2,...) [c, ceq] = feval(nonlcon,x,P1,P2,...) [c,ceq,K1,K2,...,Kn,s]= ... feval(seminfcon,x,s,P1,P2,...) 使用该特点，赋给不同的参数，则相同的 fun 函数（或 nnolcon 函数或 seminfcon 函数）可以求解一系列相似的问题，而避免使用全局变量	fgoalattain, fminbnd, fmincon, fminimax, fminsearch, fminunc, fseminf, fsolve, fzero, lsqcurvefit, lsqnonlin
seminfcon	计算非线性不等式约束、等式约束和半无限约束的函数。seminfcon 函数为 M 文件名或 MEX 文件名	Fseminf
weight	控制对象未达到或超出的加权向量	Fgoalattain
xdata, ydata	拟合方程的输入数据 ***xdata*** 和测量输出数据 ***ydata***	Lsqcurvefit
x0	初始点（标量、向量或矩阵）。（对于 fzero 函数，***x***0 还可以是一个代表包含 0 的区间的二元素向量）	All functions except fminbnd
x1, x2	函数最小化的区间	Fminbnd

表 15-8　输出变量表

变　　量	描　　述	调 用 函 数
attainfactor	解 ***x*** 处的达到因子	fgoalattain
exitflag	退出条件	
fval	解 ***x*** 处的目标函数值	fgoalattain, fminbnd, fmincon, fminimax, fminsearch, fminunc, fseminf, fsolve, fzero, linprog, quadprog
grad	解 ***x*** 处 fun 函数的梯度值	fmincon, fminunc
hessian	解 ***x*** 处 fun 函数的 Hess 矩阵值	fmincon, fminunc
jacobian	解 ***x*** 处 fun 函数的 Jacob 矩阵值	lsqcurvefit, lsqnonlin, fsolve
lambda	解 ***x*** 处的拉格朗日乘子。lambda 为一结构，它的每个字段对应于不同的约束类型。对于结构域名，可参见不同函数的描述	fgoalattain, fmincon, fminimax, fseminf, linprog, lsqcurvefit, lsqlin, lsqnonlin, lsqnonneg, quadprog
maxfval	解 ***x*** 处的函数最大值	fminimax
output	包含优化结果信息的输出结构	All functions
residual	解 ***x*** 处的残差值	lsqcurvefit, lsqlin, lsqnonlin, lsqnonneg
resnorm	解 ***x*** 处残差的卡方范数	lsqcurvefit, lsqlin, lsqnonlin, lsqnonneg
x	由优化函数求得的解。如果 exitflag > 0，则 ***x*** 为解；否则，***x*** 不是最终解，它是迭代终止时优化过程的值	All functions

表 15-9 描述优化参数结构 options 中的元素。其中，列标签 L, M 和 B 的意义如下：

L——只适用于大型问题的参数；

M——只适用于中型问题的参数；

B——对大型问题和中型问题都适用的参数。

表 15-9　优化参数表

参　数　名	描　　述	L，M，B	调 用 函 数
DerivativeCheck	对自定义的解析导数（梯度或雅可比矩阵）与有限差分导数进行比较	M	fgoalattain, fmincon, fminimax, fminunc, fseminf, fsolve, lsqcurvefit, lsqnonlin
Diagnostics	打印进行最小化或求解的诊断信息	B	除了 fminbnd、fminsearch、fzero 和 lsqnonneg 以外的所有函数
DiffMaxChange	有限差分求导的变量的最大变化	M	fgoalattain, fmincon, fminimax, fminunc, fseminf, fsolve, lsqcurvefit, lsqnonlin
DiffMinChange	有限差分求导变量的最小变化		fgoalattain, fmincon, fminimax, fminunc, fseminf, fsolve, lsqcurvefit, lsqnonlin
Display	显示水平。值为'off'时，不显示输出；值为'iter'时，显示每次迭代的信息；值为'final'时，只显示最终结果	B	全部函数
GoalsExactAchieve	精确达到的目标个数	M	fgoalattain
GradConstr	用户定义的非线性约束的梯度	M	fgoalattain, fmincon, fminimax
GradObj	用户定义的目标函数的梯度	B	fgoalattain, fmincon, fminimax, fminunc, fseminf
Hessian	用户定义的目标函数的 Hess 矩阵	L	fmincon, fminunc
HessPattern	有限差分的 Hess 矩阵的稀疏模式	L	fmincon, fminunc
HessUpdate	Hess 更新结构	M	fminunc
Jacobian	用户定义的目标函数的 Jacob 矩阵	B	fsolve, lsqcurvefit, lsqnonlin
JacobPattern	有限差分的雅可比矩阵的稀疏模式	B	fsolve, lsqcurvefit, lsqnonlin
LargeScale	如果可能的话，使用大型算法	B	fmincon, fminunc, fsolve, linprog, lsqcurvefit, lsqlin, lsqnonlin, quadprog
LevenbergMarquardt	用 Levenberg-Marquardt 法代替 Gauss-Newton 法	M	fsolve, lsqcurvefit, lsqnonlin
LineSearchType	一维搜索算法的选择	M	fminunc, fsolve, lsqcurvefit, lsqnonlin
MaxFunEvals	允许进行函数评价的最大次数	B	fgoalattain, fminbnd, fmincon, fminimax, fminsearch, fminunc, fseminf, fsolve, lsqcurvefit, lsqnonlin
MaxIter	允许进行迭代的最大次数	B	除了 fzero 和 lsqnonneg 以外所有的函数
MaxPCGIter	允许进行 PCG 迭代的最大次数	L	fmincon, fminunc, fsolve, lsqcurvefit, lsqlin, lsqnonlin, quadprog
MeritFunction	使用多目标函数（目标达到或最大最小化）	M	fgoalattain, fminimax
MinAbsMax	最小化最坏个案例绝对值的 $F(\boldsymbol{x})$的个数	M	fminimax
PrecondBandWidth	PCG 前提的上带宽	L	fmincon, fminunc, fsolve, lsqcurvefit, lsqlin, lsqnonlin, quadprog
TolCon	违背约束的终止容限	B	fgoalattain, fmincon, fminimax, fseminf
TolFun	函数值的终止容限	B	除了 fminbnd，fzero 和 lsqnonneg 以外所有的函数
TolPCG	PCG 迭代的终止容限	L	fmincon, fminunc, fsolve, lsqcurvefit, lsqlin, lsqnonlin, quadprog
TolX	$\boldsymbol{x}$ 处的终止容限	B	除了 linprog 和 lsqlin 以外所有的函数
TypicalX	典型 $\boldsymbol{x}$ 值	L	fmincon, fminunc, fsolve, lsqcurvefit, lsqlin, lsqnonlin, quadprog

15.3　参数设置

利用 optimset 函数，可以创建和编辑参数结构；利用 optimget 函数，可以获得 options 优化参数。

1．optimget 函数

利用该函数获得 options 优化参数，其语法格式如下：

● val = optimget(options, 'param') 返回优化参数 options 中指定的参数的值。只需要用参数开头的字母来定义参数就行了。

● val = optimget(options,'param',default) 若 options 结构参数中没有定义指定参数，则返回默认值。注意，这种形式的函数主要用于其他优化函数。

下面有两个使用 optimget 函数的例子：

（1）下面的命令行将显示优化参数 options 返回到 my_options 结构中：

```
val = optimget(my_options,'Display')
```

（2）下面的命令行返回显示优化参数options到my_options结构中（就像前面的例子一样），但如果显示参数没有定义，则返回值'final':

```
optnew = optimget(my_options,'Display','final');
```

2．optimset 函数

利用该函数创建或编辑优化选项参数结构。其调用格式如下：

● options = optimset('param1', value1, 'param2', value2, ...) 创建一个称为 options 的优化选项参数，其中指定的参数具有指定值。所有未指定的参数都设置为空矩阵[]（将参数设置为[ ]表示当 options 传递给优化函数时给参数赋默认值）。赋值时只要输入参数前面的字母就行了。

● optimset 函数没有输入输出变量时，将显示一张完整的带有有效值的参数列表。

● options = optimset (with no input arguments) 创建一个选项结构 options，其中所有的元素被设置为[]。

● options = optimset(optimfun) 创建一个含有所有参数名和与优化函数 optimfun 相关的默认值的选项结构 options。

● options = optimset(oldopts, 'param1', value1,...) 创建一个 oldopts 的拷贝，用指定的数值修改参数。

● options = optimset(oldopts, newopts) 将已经存在的选项结构 oldopts 与新的选项结构 newopts 进行合并。newopts 参数中的所有元素将覆盖 oldopts 参数中的所有对应元素。

调用格式中各参数的意义可参见下面的列表。其中，{ }代表默认选项；有些参数对于不同的优化函数具有不同的默认值，所以没有选项显示在{ }中。

大型算法和中型算法都可以使用的优化参数有：

```
Diagnostics        [ on | {off} ]
Display            [ off | iter | {final} ]
GradObj            [ on | {off} ]
Jacobian           [ on | {off} ]
LargeScale         [ {on} | off ]
MaxFunEvals        [ positive integer ]
```

```
MaxIter              [ positive integer ]
TolCon               [ positive scalar ]
TolFun               [ positive scalar ]
TolX                 [ positive scalar ]
```

只用于大型算法的优化参数有：

```
Hessian              [ on | {off} ]
HessPattern          [ sparse matrix ]
JacobPattern         [ sparse matrix ]
MaxPCGIter           [ positive integer ]
PrecondBandWidth     [ positive integer | Inf ]
TolPCG               [ positive scalar | {0.1} ]
TypicalX             [ vector ]
```

只用于中型算法的优化参数有：

```
DerivativeCheck      [ on | {off} ]
DiffMaxChange        [ positive scalar | {1e-1} ]
DiffMinChange        [ positive scalar | {1e-8} ]
GoalsExactAchieve    [ positive scalar integer | {0} ]
GradConstr           [ on | {off} ]
HessUpdate           [ {bfgs} | dfp | gillmurray | steepdesc ]
LevenbergMarquardt   [ on | off ]
LineSearchType       [ cubicpoly | {quadcubic} ]
MeritFunction        [ singleobj | {multiobj} ]
MinAbsMax            [ positive scalar integer | {0} ]
```

下面是几个使用 optmset 函数的例子：

（1）下面的语句创建一个称为 options 的优化选项结构，其中显示参数设为'iter'，TolFun 参数设置为 1e-8：

```
options = optimset('Display','iter','TolFun',1e-8)
```

（2）下面的语句创建一个称为 options 的优化结构的拷贝，改变 TolX 参数的值，将新值保存到 optnew 参数中：

```
optnew = optimset(options,'TolX',1e-4);
```

（3）下面的语句返回 options 优化结构，其中包含所有的参数名和与 fminbnd 函数相关的默认值：

```
options = optimset('fminbnd')
```

（4）若只希望看到 fminbnd 函数的默认值，只需要简单地键入下面的语句就行了：

```
optimset fminbnd
```

或者输入下面的命令，其效果与上面的相同：

```
optimset('fminbnd')
```

15.4 模型输入时需要注意的问题

使用优化工具箱时，由于优化函数要求目标函数和约束条件满足一定的格式，所以需要用户在进行模型输入时注意以下几个问题。

1. 目标函数最小化

优化函数 fminbnd, fminsearch, fminunc, fmincon, fgoalattain, fminmax 和 lsqnonlin 都要求目标函数最小化，如果优化问题要求目标函数最大化，可以通过使该目标函数的负值最小化即 $-f(\boldsymbol{x})$ 最小化来实现。近似地，对于 quadprog 函数提供 $-\boldsymbol{H}$ 和 $-f$，对于 linprog 函数提供 $-f$。

2. 约束非正

优化工具箱要求非线性不等式约束的形式为 $\boldsymbol{C}_i(\boldsymbol{x})\leqslant 0$，通过对不等式取负可以达到使大于零的约束形式变为小于零的不等式约束形式的目的。例如 $\boldsymbol{C}_i(\boldsymbol{x})\geqslant 0$ 形式的约束等价于 $-\boldsymbol{C}_i(\boldsymbol{x})\leqslant 0$；$\boldsymbol{C}_i(\boldsymbol{x})\geqslant \boldsymbol{b}$ 形式的约束等价于 $-\boldsymbol{C}_i(\boldsymbol{x})+\boldsymbol{b}\leqslant 0$。

3. 避免使用全局变量

全局变量在整个程序中都可用，当程序比较大时，难免会在无意中修改全局变量的值，因而导致错误。更糟糕的是，这样的错误还很难查找。因此，编程时应该尽量避免使用全局变量。

15.5 @（函数句柄）函数

MATLAB 中可以用@函数进行函数调用。@函数返回指定 MATLAB 函数的句柄，其调用格式为：

```
handle = @function
```

利用@函数进行函数调用有下面几点好处：

① 用句柄将一个函数传递给另一个函数；

② 减少定义函数的文件个数；

③ 改进重复操作；

④ 保证函数计算的可靠性。

下面的例子为 humps 函数创建一个函数句柄，并将它指定为 fhandle 变量。

```
fhandle = @humps;
```

同样，传递句柄给另一个函数，也将传递所有变量。本例将刚刚创建的函数句柄传递给 fminbnd 函数，然后在区间[0.3, 1]上进行最小化。

```
x = fminbnd (@humps, 0.3, 1)
x =
   0.6370
```

关于@函数的更多细节，可以参见本套书第一册《MATLAB 程序设计》中第 2 章的内容。

第 16 章　无约束最优化问题

16.1　单变量最小化

16.1.1　基本数学原理

本节讨论只有一个变量时的最小化问题，即一维搜索问题。该问题在某些情况下可以直接用于求解实际问题，但大多数情况下它是作为多变量最优化方法的基础，因为进行多变量最优化要用到一维搜索算法。该问题的数学模型为：

$$\min f(x) \qquad x_1 < x < x_2$$

式中，x, x_1, 和 x_2 为标量，$f(x)$为函数，返回标量。

该问题的搜索过程可用下式表达：

$$x_{k+1} = x_k + \alpha * d$$

式中 x_k 为本次迭代的值，d 为搜索方向，α 为搜索方向上的步长参数。所以一维搜索就是要利用本次迭代的信息来构造下次迭代的条件。

求解单变量最优化问题的方法有很多种。根据目标函数是否需要求导，可以分为两类，即直接法和间接法。直接法不需要目标函数的导数，而间接法则需要用到目标函数的导数。

1．直接法

常用的一维直接法主要有消去法和近似法两种。

（1）消去法。该法利用单峰函数具有的消去性质进行反复迭代，逐渐消去不包含极小点的区间，缩小搜索区间，直到搜索区间缩小到给定的允许精度为止。一种典型的消去法为黄金分割搜索法(Golden Section Search)。黄金分割搜索法的基本思想是在单峰区间内适当插入两点，将区间分为 3 段，然后通过比较这两点函数值的大小来确定是删去最左段还是最右段，或同时删去左、右两段保留中间段。重复该过程使区间无限缩小。插入点的位置放在区间的黄金分割点及其对称点上，所以该法称为黄金分割搜索法。该法的优点是算法简单，效率较高，稳定性好。

（2）多项式近似法。该法用于目标函数比较复杂的情况。此时寻找一个与它近似的函数来代替目标函数，并用近似函数的极小点作为原函数极小点的近似。常用的近似函数为二次和三次多项式。

二次内插涉及到形如下式的二次函数数据拟合问题：

$$m_q(\alpha) = a\alpha^2 + b\alpha + c$$

其中步长极值为

$$\alpha^* = \frac{-b}{2a}$$

然后只要利用 3 个梯度或函数方程组就可以确定系数 a 和 b，从而可以确定α^*。得到该值以后，进行搜索区间的收缩。在缩短的新区间中，重新安排 3 点求出下一次的近似极小点

α^*，如此迭代下去，直到满足终止准则为止。其迭代公式为

$$x_{k+1}=\frac{1}{2}\frac{\beta_{23}f(x_1)+\beta_{31}\ f(x_2)+\beta_{12}f(x_3)}{\gamma_{23}f(x_1)+\gamma_{31}f(x_2)+\gamma_{12}f(x_3)}$$

式中

$$\beta_{ij}=x_i^2-x_j^2，\ \gamma_{ij}=x_i-x_j$$

二次插值法的计算速度比黄金分割搜索法的快，但是对于一些强烈扭曲或可能多峰的函数，该法的收敛速度会变得很慢，甚至失败。

2．间接法

间接法需要计算目标函数的导数，优点是计算速度很快。常见的间接法包括牛顿切线法、对分法、割线法和三次插值多项式近似法等。优化工具箱中用得较多的是三次插值法。

三次插值的基本思想与二次插值的一致，它是用 4 个已知点构造一个三次多项式 $P3(x)$，用它逼近函数 $f(x)$，以 $P3(x)$的极小点作为 $f(x)$的近似极小点。一般地讲，三次插值法比二次插值法的收敛速度要快些，但每次迭代需要计算两个导数值。

三次插值法的迭代公式为

$$x_{k+1}=x_2-(x_2-x_1)\frac{\nabla f(x_2)+\beta_2-\beta_1}{\nabla f(x_2)-\nabla f(x_1)+2\beta_2}$$

式中

$$\beta_1=\nabla f(x_1)+\nabla f(x_2)-3\frac{f(x_1)-f(x_2)}{x_1-x_2}$$

$$\beta_2=(\beta_1^2-\nabla f(x_1)\nabla f(x_2))^{1/2}$$

如果函数的导数容易求得，一般来说首先考虑使用三次插值法，因为它具有较高的效率。对于只需要计算函数值的方法中，二次插值法是一个很好的方法，它的收敛速度较快，在极小点所在区间较小时尤其如此。黄金分割法则是一种十分稳定的方法，并且计算简单。由于以上原因，MATLAB 优化工具箱中用得较多的方法是二次插值法、三次插值法以及二次、三次混合插值法和黄金分割法。

16.1.2 有关函数介绍

1．fminbnd 函数

利用该函数找到固定区间内单变量函数的最小值。调用格式为：

- x = fminbnd(fun,x1,x2)返回区间{x1，x2}上 fun 参数描述的标量函数的最小值 *x*。
- x = fminbnd(fun,x1,x2,options)用 options 参数指定的优化参数进行最小化。
- x = fminbnd(fun,x1,x2,options,P1,P2,...)提供另外的参数 P1，P2 等，传输给目标函数 fun。如果没有设置 options 选项，则令 options=[ ]。
- [x,fval] = fminbnd(...)返回解 x 处目标函数的值。
- [x,fval,exitflag] = fminbnd(...)返回 exitflag 值描述 fminbnd 函数的退出条件。
- [x,fval,exitflag,output] = fminbnd(...)返回包含优化信息的结构输出。

与 fminbnd 函数相关的细节内容包含在 fun，options，exitflag 和 output 等参数中，如表 16-1 所示。

表 16-1　参数描述表

参　数	描　　述
fun	需要最小化的目标函数。Fun 函数需要输入标量参数 x，返回 x 处的目标函数标量值 f。可以将 fun 函数指定为命令行，如 x = fminbnd(inline('sin(x*x)'),x0) 同样，fun 参数可以是一个包含函数名的字符串。对应的函数可以是 M 文件、内部函数或 MEX 文件。若 fun='myfun'，则 M 文件函数 myfun.m 必须有下面的形式 function f = myfun(x) f = ...　　　　　%计算 x 处的函数值
options	优化参数选项。可以用 optimset 函数设置或改变这些参数的值。options 参数有以下几个选项： Display　显示的水平。选择'off'，不显示输出；选择'iter'，显示每一步迭代过程的输出；选择'final'，显示最终结果 MaxFunEvals　函数评价的最大允许次数 MaxIter　最大允许迭代次数 TolX　x 处的终止容限
exitflag	描述退出条件： >0　表示目标函数收敛于解 x 处 0　表示已经达到函数评价或迭代的最大次数 <0　表示目标函数不收敛
output	该参数包含下列优化信息： output.iterations　迭代次数 output.algorithm　所采用的算法 output.funcCount　函数评价次数

注意：

- 目标函数必须是连续的；
- fminbnd 函数可能只给出局部最优解；
- 当问题的解位于区间边界上时，fminbnd 函数的收敛速度常常很慢。此时，fmincon 函数的计算速度更快，计算精度更高；
- fminbnd 函数只用于实数变量。

2．应用实例

【例 16-1】　对边长为 3m 的正方形铁板，在 4 个角处剪去相等的正方形以制成方形无盖水槽，问如何剪法使水槽的容积最大？

假设剪去的正方形的边长为 x，则水槽的容积为

$$f(x)=(3-2x)^2x$$

现在要求在区间（0，1.5）上确定一个 x，使 $f(x)$ 最大化。因为优化工具箱中要求目标函数最小化，所以需要对目标函数进行转换，即要求 $-f(x)$ 最小化。

首先编写 M 文件 opt16_1o.m:

```
function f = myfun(x)
f = - (3-2*x).^2 * x;
```

然后调用 fminbnd 函数：

```
x = fminbnd(@opt16_1o,0,1.5)
```

得到问题的解：

```
x =
    0.5000
```

即剪掉的正方形的边长为 0.5m 时水槽的容积最大。

水槽的最大容积计算：

```
y = optim2(x)
y =
    -2.0000
```

所以水槽的最大容积为 2.0000m³。

16.2　无约束非线性规划问题

16.2.1　基本数学原理

无约束最优化问题在实际应用中也比较常见，如工程中常见的参数反演问题。另外，许多有约束最优化问题可以转化为无约束最优化问题进行求解。

求解无约束最优化问题的方法主要有两类，即直接搜索法（Direct search method）和梯度法（Gradient method）。

直接搜索法适用于目标函数高度非线性，没有导数或导数很难计算的情况。由于实际工程中很多问题都是非线性的，故直接搜索法不失为一种有效的解决办法。常用的直接搜索法为单纯形法，此外还有 Hooke-Jeeves 搜索法、Pavell 共轭方向法等，其缺点是收敛速度慢。

在函数的导数可求的情况下，梯度法是一种更优的方法。该法利用函数的梯度（一阶导数）和 Hess 矩阵（二阶导数）构造算法，可以获得更快的收敛速度。函数 $f(x)$的负梯度方向 $-\nabla f(x)$即反映了函数的最大下降方向。当搜索方向取为负梯度方向时称为最速下降法。当需要最小化的函数有一狭长的谷形值域时，该法的效率很低，如 Rosenbrock 函数

$$f(\boldsymbol{x})=100(x_1-x_2^2)^2+(1-x_1)^2$$

它的最小值解为 $\boldsymbol{x}$=[1,1]，最小值为 $f(\boldsymbol{x})$=0。图 16-1 是该函数的等值线图，图中还显示了从初值[-1.9，2]出发向最小值前进的路径。迭代 1000 次以后终止，此时距最小值仍有相当长的距离。图中的黑色区域是该法在谷的两侧不断进行“之”字形搜索形成的。

这种类型的函数又称为香蕉函数。

常见的梯度法有最速下降法、Newton 法、Marquart 法、共轭梯度法和拟牛顿法（Quasi-Newton method）等。

在所有这些方法中，用得最多的是拟牛顿法，这些方法在每次迭代过程中建立曲率信息，构成下式得二次模型问题：

$$\min_{X} \frac{1}{2}\boldsymbol{X}^T\boldsymbol{H}\boldsymbol{X}+\boldsymbol{C}^T\boldsymbol{X}+b$$

式中，Hess 矩阵 $\boldsymbol{H}$ 为一正定对称矩阵，$\boldsymbol{C}$ 为常数向量，b 为常数。对 $\boldsymbol{x}$ 求偏导数可以获得问题的最优解

$$\nabla f(\boldsymbol{x}^*)=\boldsymbol{H}\boldsymbol{x}^*+\boldsymbol{C}=0$$

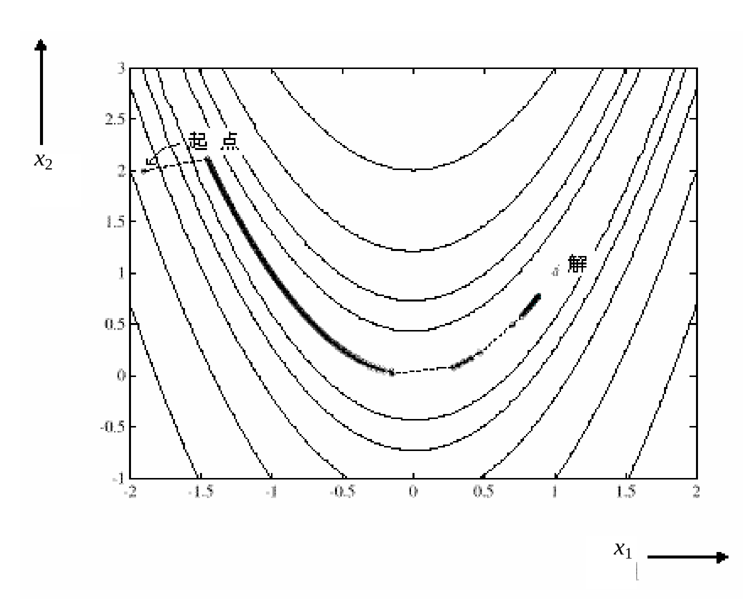


图 16-1　香蕉函数的等值线图及最速下降法的搜索路径

解 $\boldsymbol{x}^*$ 可写成：

$$\boldsymbol{x}^* = -\boldsymbol{H}^{-1}\boldsymbol{C}$$

拟牛顿法包括两个阶段，即确定搜索方向和一维搜索阶段

1．Hess 矩阵的更新

牛顿法由于需要多次计算 Hess 矩阵，故计算量很大。而拟牛顿法则通过构建一个 Hess 矩阵的近似矩阵来避开这个问题。

在优化工具箱中，通过将 options 参数 HessUpdate 设置为 BFGS 或 DFP 来决定搜索方向。当 Hess 矩阵 $\boldsymbol{H}$ 始终保持正定时，搜索方向就总是保持为下降方向。

构建 Hess 矩阵的方法很多。对于求解一般问题，Broyden, Fletcher, Goldfarb 和 Shanno 的方法（简称 BFGS 法）是最有效的。BFGS 法的计算公式为

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \frac{\boldsymbol{q}_k \boldsymbol{q}_k^T}{\boldsymbol{q}_k^T \boldsymbol{S}_k} - \frac{\boldsymbol{H}_k^T \boldsymbol{S}_k^T \boldsymbol{S}_k \boldsymbol{H}_k}{\boldsymbol{S}_k^T \boldsymbol{H}_k \boldsymbol{S}_k}$$

式中：$\boldsymbol{S}_k = \boldsymbol{x}_{k+1} - \boldsymbol{x}_k$，$\boldsymbol{q}_k = \nabla f(\boldsymbol{x}_{k+1}) - \nabla f(\boldsymbol{x}_k)$

作为初值，$\boldsymbol{H}_0$ 可以设为任意对称正定矩阵。

另一个有名的构造近似 Hess 矩阵的方法是 DFP（Daridon-Fletcher-Powell）法。该法的计算公式与 BFGS 法的形式一样，只是 $\boldsymbol{q}_k$ 替换为 $\boldsymbol{S}_k$。

梯度信息可以用解析方法得到，也可以用有限差分方法通过求偏导数得到。

在每一个主要的迭代过程中，在下式所示的方向上进行一维搜索：

$$d = -\boldsymbol{H}_k^{-1} \cdot \nabla f(\boldsymbol{x}_k)$$

图 16-2 演示了用拟牛顿法时 Rosenbrock 函数的求解路径。可见，利用该法，只需要 140 次迭代就可以达到最小值解，比前面利用最速下降法要快得多。

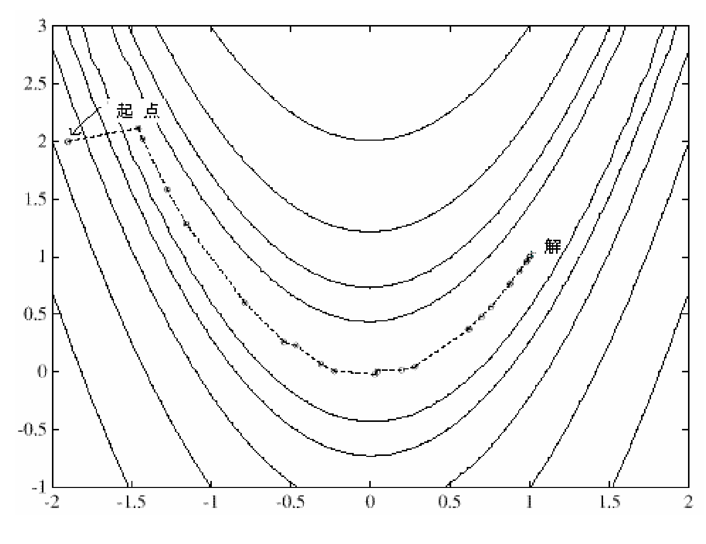


图 16-2　拟牛顿法的搜索路径

2．一维搜索

工具箱中有两套方案进行一维搜索。当梯度值可以直接得到时，用三次插值的方法进行一维搜索，当梯度值不能直接得到时，采用二次、三次混合插值法。

16.2.2　有关函数介绍

1．fminunc 函数

用该函数求多变量无约束函数的最小值。多变量无约束函数的数学模型为：

$$\min_{x} f(\boldsymbol{x})$$

式中，$\boldsymbol{x}$ 为一向量，$f(\boldsymbol{x})$为一函数，返回标量。

fminunc 函数的调用格式如下：

- fminunc 给定初值，求多变量标量函数的最小值。常用于无约束非线性最优化问题。
- x = fminunc(fun,x0)给定初值 x0，求 fun 函数的局部极小点 x。x0 可以是标量、向量或矩阵。
- x = fminunc(fun,x0,options)用 options 参数中指定的优化参数进行最小化。
- x = fminunc(fun,x0,options,P1,P2,...)将问题参数 P1、P2 等直接输给目标函数 fun，将 options 参数设置为空矩阵，作为 options 参数的默认值。
- [x,fval] = fminunc(...)将解 x 处目标函数的值返回到 fval 参数中。
- [x,fval,exitflag] = fminunc(...)返回 exitflag 值，描述函数的输出条件。
- [x,fval,exitflag,output] = fminunc(...)返回包含优化信息的结构输出。
- [x,fval,exitflag,output,grad] = fminunc(...)将解 x 处 fun 函数的梯度值返回到 grad 参数中。
- [x,fval,exitflag,output,grad,hessian] = fminunc(...)将解 x 处目标函数的 Hessian 矩阵信息返回到 hessian 参数中。

表 16-2 中包括各输入/输出变量的描述。

表 16-2　输入/输出变量描述表

变　量	描　　述
fun	为目标函数，即需要最小化的目标函数。fun 函数需要输入标量参数 x，返回 x 处的目标函数标量值 f。可以将 fun 函数指定为命令行，如 x = fminbnd(inline('sin(x*x)'),x0) 同样，fun 参数可以是一个包含函数名的字符串。对应的函数可以是 M 文件、内部函数或 MEX 文件。若 fun='myfun'，则 M 文件函数 myfun.m 必须有下面的形式： function f = myfun(x) f = ...　　　　　%计算 x 处的函数值 若 fun 函数的梯度可以算得，且 options.GradObj 设为'on'（用下式设定）， options = optimset('GradObj','on') 则 fun 函数必须返回解 x 处的梯度向量 ***g*** 到第 2 个输出变量中去。注意，当被调用的 fun 函数只需要一个输出变量时（如算法只需要目标函数的值而不需要其梯度值时），可以通过核对 nargout 的值来避免计算梯度值 function [f,g] = myfun(x) f = ...　　　　%计算 x 处的函数值 if nargout > 1　%调用 fun 函数并要求有两个输出变量 　g = ...　　　%计算 x 处的梯度值 end 若 Hess 矩阵也可以求得，并且 options.Hessian 设为'on',即, options = optimset('Hessian','on') 则 fun 函数必须返回解 x 处的 Hess 对称矩阵 ***H*** 到第 3 个输出变量中去。注意，当被调用的 fun 函数只需要一个或两个输出变量时（如算法只需要目标函数的值 ***f*** 和梯度值 ***g*** 而不需要 Hess 矩阵 ***H*** 时），可以通过核对 nargout 的值以避免计算 Hess 矩阵 function [f,g,H] = myfun(x) f = ...　　　　% 计算 x 处的函数值 if nargout > 1　% 调用 fun 函数并要求有两个输出变量 　g = ...　　　% 计算 x 处的梯度值 　if nargout > 2 　H = ...　　　% 计算 x 处的 Hess 矩阵 end
options	优化参数选项。可以通过 optimset 函数设置或改变这些参数。其中有的参数适用于所有的优化算法，有的则只适用于大型优化问题，另外一些则只适用于中型问题 首先描述适用于大型问题的选项。这仅仅是一个参考，因为使用大型问题算法有一些条件。对于 fminunc 函数来说，必须提供梯度信息 ● LargeScale　当设为'on'时使用大型算法，若设为'off'则使用中型问题的算法 适用于大型和中型算法的参数： ● Diagnostics　打印最小化函数的诊断信息 ● Display　显示水平。选择'off'，不显示输出；选择'iter'，显示每一步迭代过程的输出；选择'final'，显示最终结果。打印最小化函数的诊断信息 ● GradObj　用户定义的目标函数的梯度。对于大型问题此参数是必选的，对于中型问题则是可选项 ● MaxFunEvals　函数评价的最大次数 ● MaxIter　最大允许迭代次数 ● TolFun　函数值的终止容限 ● TolX　x 处的终止容限 只用于大型算法的参数： ● Hessian　用户定义的目标函数的 Hess 矩阵 ● HessPattern　用于有限差分的 Hess 矩阵的稀疏形式。若不方便求 fun 函数的稀疏 Hess 矩阵 ***H***，可以用梯度的有限差分获得的 ***H*** 的稀疏结构（如非零值的位置等）得到近似的 Hess 矩阵 ***H***。若连矩阵的稀疏结构都不知道，则可以将 HessPattern 设为密集矩阵，在每一次迭代过程中，都将进行密集矩阵的有限差分近似（这是默认设置）。这将非常麻烦，所以花一些力气得到 Hess 矩阵的稀疏结构还是值得的 ● MaxPCGIter　PCG 迭代的最大次数

续表

options	● PrecondBandWidth　PCG 前处理的上带宽，默认时为零。对于有些问题，增加带宽可以减少迭代次数 ● TolPCG　PCG 迭代的终止容限 ● TypicalX　典型 x 值 只用于中型算法的参数： ● DerivativeCheck　对用户提供的导数和有限差分求出的导数进行对比 ● DiffMaxChange　变量有限差分梯度的最大变化 ● DiffMinChange　变量有限差分梯度的最小变化 ● LineSearchType　一维搜索算法的选择
exitflag	描述退出条件： ● >0　表示目标函数收敛于解 x 处 ● 0　表示已经达到函数评价或迭代的最大次数 ● <0　表示目标函数不收敛
output	该参数包含下列优化信息： ● output.iterations　迭代次数 ● output.algorithm　所采用的算法 ● output.funcCount　函数评价次数 ● output.cgiterations　PCG 迭代次数（只适用于大型规划问题） ● output.stepsize　最终步长的大小（只用于中型问题） ● output.firstorderopt　一阶优化的度量：解 x 处梯度的范数

对规模不同的优化问题，fminunc 函数使用不同的优化算法。

（1）大型优化算法

若用户在 fun 函数中提供梯度信息，则默认时函数将选择大型优化算法。该算法是基于内部映射牛顿法的子空间置信域法。计算中的每一次迭代都涉及到用 PCG 法求解大型线性系统得到的近似解。

（2）中型优化算法

此时 fminunc 函数的参数 options.LargeScale 设置为'off'。该算法采用的是基于二次和三次混合插值一维搜索法的 BFGS 拟牛顿法。该法通过 BFGS 公式来更新 Hess 矩阵。通过将 HessUpdate 参数设置为'dfp'，可以用 DFP 公式来求得 Hess 矩阵逆的近似。通过将 HessUpdate 参数设置为'steepdesc'，可以用最速下降法来更新 Hess 矩阵。但一般不建议使用最速下降法。

当 options.LineSearchType 设置为'quadcubic'时，默认一维搜索算法为二次和三次混合插值法。将 options.LineSearchType 设置为'cubicpoly'时，将采用三次插值法。第 2 种方法需要的目标函数计算次数更少，但梯度的计算次数更多。这样，如果提供了梯度信息，或者能较容易地算得，则三次插值法是更佳的选择。

注意：

① 对于求解平方和的问题，fminunc 函数不是最好的选择，用 lsqnonlin 函数效果更佳。

② 使用大型方法时，必须通过将 options.GradObj 设置为'on'来提供梯度信息，否则将给出警告消息。

③ 目标函数必须是连续的。fminunc 函数有时会给出局部最优解。

④ fminunc 函数只对实数进行优化，即 x 必须为实数，而且 $f(\boldsymbol{x})$必须返回实数。当 $\boldsymbol{x}$ 为复数时，必须将它分解为实部和虚部。

⑤ 在使用大型算法时，用户必须在 fun 函数中提供梯度（options 参数中 GradObj 属性必须设置为'on'）。

目前，若在 fun 函数中提供了解析梯度，则 options 参数 DerivativeCheck 不能用于大型算法以比较解析梯度和有限差分梯度。通过将 options 参数的 MaxIter 属性设置为 0 以用中型方法核对导数。然后重新用大型方法求解问题。

【例 16-2】 将下列函数最小化

```
f(x) = 3*x12 + 2*x1*x2 + x22
```

首先创建 M 文件 myfun.m：

```
function f = myfun(x)
f = 3*x(1)^2 + 2*x(1)*x(2) + x(2)^2;    % 目标函数
```

然后调用 fminunc 函数求[1，1]附近'myfun'函数的最小值。

```
x0 = [1,1];
[x,fval] = fminunc(@myfun,x0)
```

经过 12 次迭代以后，返回解 ***x*** 和 ***x*** 处的函数值 fval。

```
x =
    1.0e-008 *
   -0.7914    0.2260
fval =
    1.5722e-016
```

下面用提供的梯度 *g* 使函数最小化，修改 M 文件如下：

```
function [f,g] = myfun(x)
f = 3*x(1)^2 + 2*x(1)*x(2) + x(2)^2;    % 目标函数
if nargout > 1
   g(1) = 6*x(1)+2*x(2);
   g(2) = 2*x(1)+2*x(2);
end
```

下面通过将优化选项结构 options.GradObj 设置为'on'来得到梯度值。

```
options = optimset('GradObj','on');
x0 = [1,1];
[x,fval] = fminunc(@myfun,x0,options)
```

经过数次迭代以后，返回解 *x* 和 *x* 处的函数值 fval2。

```
x =
    1.0e-015 *
   -0.6661         0
fval2 =
    1.3312e-030
```

2. fminsearch 函数

利用 fminsearch 函数求解多变量无约束函数的最小值，其调用格式如下：

- fminsearch 求解多变量无约束函数的最小值。该函数常用于无约束非线性最优化问题。
- x = fminsearch(fun,x0) 初值为 x0，求 fun 函数的局部极小点 x。x0 可以是标量、向量或矩阵。
- x = fminsearch(fun,x0,options)用 options 参数指定的优化参数进行最小化。
- x = fminsearch(fun,x0,options,P1,P2,...) 将问题参数 p1，p2 等直接输给目标函数 fun，将 options 参数设置为空矩阵，作为 options 参数的默认值。
- [x,fval] = fminsearch(...)将 x 处的目标函数值返回到 fval 参数中。

- [x,fval,exitflag] = fminsearch(...)返回 exitflag 值，描述函数的退出条件。
- [x,fval,exitflag,output] = fminsearch(...)返回包含优化信息的输出参数 output。

fminsearch 函数使用单纯形法进行计算。

对于求解二次以上的问题，fminsearch 函数比 fminunc 函数有效。但是，当问题为高度非线性时，fminsearch 函数更具稳健性。

注意：

应用 fminsearch 函数可能会得到局部最优解。fminsearch 函数只对实数进行最小化，即 x 必须由实数组成，$f(x)$函数必须返回实数。如果 x 为复数，则必须将它分为实数部和虚数部两部分。

另外，fminsearch 函数不适合求解平方和问题，用 lsqnonlin 函数更好一些。

【例 16-3】 使一维函数 $f(x) = \sin(x) + 3$ 最小化。

首先创建 M 文件 myfun.m：

```
function f = myfun(x)
f = sin(x) + 3;
```

然后调用 fminsearch 函数求 2 附近函数的最小值。

```
x = fminsearch(@myfun,2)
```

下面使用命令行使该函数最小化：

```
f = inline('sin(x)+3');
x = fminsearch(f,2);
```

第 17 章　有约束最优化问题

17.1　线性规划

线性规划是处理线性目标函数和线性约束的一种较为成熟的方法，目前已经广泛应用于军事、经济、工业、农业、教育、商业和社会科学等许多方面。

线性规划的求解方法主要是单纯形法（Simple Method）。该法由 Dantzig 于 1947 年提出，以后经过多次改进。单纯形法是一种迭代算法，它从所有基本可行解的一个较小部分中通过迭代过程选出最优解。其迭代过程的一般描述如下。

① 将线性规划化为典范形式，从而可以得到一个初始基本可行解 $\boldsymbol{x}^{(0)}$(初始顶点)，将它作为迭代过程的出发点，其目标值为 $z(\boldsymbol{x}^{(0)})$。

② 寻找一个基本可行解 $\boldsymbol{x}^{(1)}$，使 $z(\boldsymbol{x}^{(1)})\leqslant z(\boldsymbol{x}^{(0)})$。方法是通过消去法将产生 $\boldsymbol{x}^{(0)}$的典范形式化为产生 $x^{(1)}$的典范形式。

③ 继续寻找较好的基本可行解 $\boldsymbol{x}^{(2)}$，$\boldsymbol{x}^{(3)}$，…，使目标函数值不断改进，即 $z(\boldsymbol{x}^{(1)})\geqslant z(\boldsymbol{x}^{(2)})\geqslant z(\boldsymbol{x}^{(3)})\geqslant\cdots$。当某个基本可行解再也不能被其他基本可行解改进时，它就是所求的最优解。

MATLAB 优化工具箱中采用的是投影法，它是单纯形法的一种变种。

17.1.1　基本数学原理

线性规划问题的标准形式是

$$\begin{cases}\min z=c_1x_1+c_2x_2+\cdots+c_nx_n\\ a_{11}x_1+a_{12}x_2+\cdots+a_{1n}x_n=b_1\\ a_{21}x_1+a_{22}x_2+\cdots+a_{2n}x_n=b_2\\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\\ a_{m1}x_1+a_{m2}x_2+\cdots+a_{mn}x_n=b_m\\ x_1,x_2,\cdots,x_n\geqslant 0\end{cases}$$

或

$$\begin{cases}\min z=\sum_{j=1}^{n}c_jx_j\\ \sum_{j=1}^{n}a_{ij}x_j=b_i, \quad i=1,2,\ldots,m\\ x_j\geqslant 0, \quad j=1,2,\ldots,n\end{cases}$$

写成矩阵形式为

$$\begin{cases}\min \boldsymbol{z}=\boldsymbol{CX}\\ \boldsymbol{AX}=\boldsymbol{b}\\ \boldsymbol{X}\geqslant\boldsymbol{0}\end{cases}$$

式中，$\boldsymbol{0}$ 为 n 维列向量。

线性规划的标准形式要求使目标函数最小化，约束条件取等式，变量 b 非负。不符合这几个条件的线性模型要首先转化成标准形式。

17.1.2　有关函数介绍

在 MATLAB 工具箱中，可用 linprog 函数求解线性规划问题。

假设线性规划问题的数学模型为：

$$\min_{x} \boldsymbol{f}^T \boldsymbol{x}$$
$$\boldsymbol{A} \cdot \boldsymbol{x} \leqslant \boldsymbol{b}$$
$$\boldsymbol{Aeq} \cdot \boldsymbol{x} = \boldsymbol{beq}$$
$$\boldsymbol{lb} \leqslant \boldsymbol{x} \leqslant \boldsymbol{ub}$$

式中 ***f***, ***x***, ***b***, ***beq***, ***lb*** 和 ***ub*** 为向量，***A*** 和 ***Aeq*** 为矩阵。

linprog 函数的调用格式如下：

- x = linprog (f, A, b) 求解问题 min f'*x，约束条件为 A*x <= b。
- x = linprog (f, A, b, Aeq, beq) 求解上面的问题，但增加等式约束，即 Aeq*x = beq。若没有不等式存在，则令 A=[ ]、b=[ ]。
- x = linprog (f, A, b, Aeq, beq, lb, ub) 定义设计变量 x 的下界 lb 和上界 ub，使得 x 始终在该范围内。若没有等式约束，令 Aeq=[ ]、beq=[ ]。
- x = linprog(f, A, b, Aeq, beq, lb, ub, x0) 设置初值为 x0。该选项只适用于中型问题，默认时大型算法将忽略初值。
- x = linprog (f, A, b, Aeq, beq, lb, ub, x0, options) 用 options 指定的优化参数进行最小化。
- [x, fval] = linprog (...) 返回解 x 处的目标函数值 fval。
- [x, lambda, exitflag] = linprog (...)返回 exitflag 值，描述函数计算的退出条件。
- [x, lambda, exitflag, output] = linprog (...) 返回包含优化信息的输出变量 output。
- [x,fval, exitflag, output, lambda] = linprog (...) 将解 x 处的拉格朗日乘子返回到 lambda 参数中。

调用格式中，lambda 参数是解 x 处的拉格朗日乘子。它有以下一些属性：

- lambda.lower——lambda 的下界。
- lambda.upper——lambda 的上界。
- lambda.ineqlin——lambda 的线性不等式。
- lambda.eqlin——lambda 的线性等式。

其他参数意义可参见表 15-7 和表 15-8。

根据问题规模的不同，linprog 函数使用不同的算法：

- 大型优化算法——大型优化算法采用的是 LIPSOL 法。该法在进行迭代计算之前首先要进行一系列的预处理。
- 中型优化算法——linprog 函数使用的是投影法，就象 quadprog 函数的算法一样。linprog 函数使用的是一种活动集方法，是线性规划中单纯形法的变种。它通过求解另一个线性规划问题来找到初始可行解。

对于大型算法，算法的第 1 步涉及到一些约束条件的预处理问题。有些问题可能导致 linprog 函数退出，并显示不可行的消息。在本例中，exitflag 参数将被设为负值以表示优化失败。

若 Aeq 参数中某行的所有元素都为零，但 Beq 参数中对应的元素不为零，则显示以下退出

消息：

Exiting due to infeasibility: an all zero row in the constraint matrix does not have a zero in corresponding right hand side entry.

若 x 的某一个元素没在界内，则给出以下退出消息：

Exiting due to infeasibility: objective f'*x is unbounded below.

若 Aeq 参数的某一行中只有一个非零值，则 x 中的相关值称为奇异变量。这里，*x* 中该成分的值可以用 Aeq 和 Beq 算得。若算得的值与另一个约束条件相矛盾，则给出以下退出消息：

Exiting due to infeasibility: singleton variables in equality constraints are not feasible.

若奇异变量可以求解但其解超出上界或下界，则给出以下退出消息：

Exiting due to infeasibility: singleton variables in the equality constraints are not within bounds.

注意：预处理步骤是累加的。例如，即使约束矩阵开始不含有元素全为零的行，其他预处理步骤也会引起某行元素全为零。

一旦预处理结束，将进行迭代计算，直到满足终止准则。若迭代的残差在增加而不是减小，或者残差不增加也不减小，则分别给出下面两条终止消息：

One or more of the residuals, duality gap, or total relative error has grown 100000 times greater than its minimum value so far:

或者

One or more of the residuals, duality gap, or total relative error has stalled:

对于中型优化问题，当解不可行时，linprog 函数给出下面的警告消息：

Warning: The constraints are overly stringent;there is no feasible solution.

这里，linprog 函数给出一个结果，使约束矛盾的最坏程度变到最小。

当等式约束不协调时，linprog 函数给出警告消息：

Warning: The equality constraints are overly stringent; there is no feasible solution.

超出边界的解给出的警告消息是：

Warning: The solution is unbounded and at infinity;the constraints are not restrictive enough.

这里，linprog 函数返回 x 的值，该值满足约束条件。

另外，对于中型优化问题，显示水平参数只能使用'off'和'final'，进行迭代输出的'iter'属性不可用。

17.1.3 应用实例

【例 17-1】 生产决策问题。某厂生产甲、乙两种产品，已知制成一吨产品甲需用资源 *A* 3 吨，资源 *B* 4m^3；制成一吨产品乙需用资源 *A* 2 吨，资源 *B* 6m^3，资源 *C* 7 个单位。若一吨产品甲和乙的经济价值分别为 7 万元和 5 万元，三种资源的限制量分别为 90 吨、200m^3 和 210 个单位。试决定应生产这两种产品各多少吨才能使创造的总经济价值最高？

令生产产品甲的数量为 x_1，生产产品乙的数量为 x_2。由题意可以建立下面的模型：

$$\begin{cases} \max z = 7x_1 + 5x_2 \\ 3x_1 + 2x_2 \leqslant 90 \\ 4x_1 + 6x_2 \leqslant 200 \\ 7x_2 \leqslant 210 \\ x_1 \geqslant 0, x_2 \geqslant 0 \end{cases}$$

该模型要求使目标函数最大化，需要按照 MATLAB 的要求进行转换，即目标函数为

$$\min \boldsymbol{z} = -7x_1 - 5x_2$$

首先输入下列系数：

```
f = [-7; -5];
A = [3 2
     4 6
     0 7];
b = [90; 200; 210];
lb = zeros(2,1);
```

然后调用 linprog 函数：

```
[x,fval,exitflag,output,lambda] = linprog(f,A,b,[ ],[ ],lb)
x =
   14.0000
   24.0000
fval =
   -218.0000
exitflag =
     1
output =
      iterations: 5
    cgiterations: 0
       algorithm: 'lipsol'
lambda =
    ineqlin: [3x1 double]
      eqlin: [0x1 double]
      upper: [2x1 double]
      lower: [2x1 double]
```

由上可知，生产甲种产品 14 吨、乙种产品 24 吨可使创建的总经济价值最高。最高经济价值为 218 万元。exitflag=1 表示过程正常收敛于解 x 处。

【例 17-2】 投资问题。某单位有一批资金用于 4 个工程项目的投资，各工程项目所得到的净收益（投入资金的百分比）如表 17-1 所示。

表 17-1　工程项目收益表

工 程 项 目	*A*	*B*	*C*	*D*
收益（%）	15	10	8	12

由于某种原因，决定用于项目 A 的投资不大于其他各项投资之和；而用于项目 B 和 C 的投资要大于项目 D 的投资。试确定使该单位收益最大的投资分配方案。

用 x_1、x_2、x_3 和 x_4 分别代表用于项目 A，B，C 和 D 的投资百分数，由于各项目的投资百分数之和必须等于 100%，所以

$$x_1+x_2+x_3+x_4=1$$

据题意，可以建立下面的数学模型：

$$\begin{cases} \max z = 0.15x_1 + 0.1x_2 + 0.08x_3 + 0.12x_4 \\ x_1 - x_2 - x_3 - x_4 \leqslant 0 \\ x_2 + x_3 - x_4 \geqslant 0 \\ x_1 + x_2 + x_3 + x_4 = 1 \\ x_j \geqslant 0, j = 1, 2, ..., 4 \end{cases}$$

将它转换为标准形式：

$$\begin{cases} \min z = \quad 0.15x_1 - 0.1x_2 - 0.08x_3 - 0.12x_4 \\ x_1 - x_2 - x_3 - x_4 \leqslant 0 \\ x_2 - x_3 + x_4 \leqslant 0 \\ x_1 + x_2 + x_3 + x_4 = 1 \\ x_j \geqslant 0, j = 1, 2, ..., 4 \end{cases}$$

下面进行求解。

首先输入下列系数：

```
f = [-0.15; -0.1; -0.08; -0.12];
A = [1 -1 -1 -1
     0 -1 -1 1];
b = [0; 0];
Aeq=[1 1 1 1];
beq=[1];
lb = zeros(4,1);
```

然后调用 linprog 函数：

```
[x,fval,exitflag,output,lambda] = linprog(f,A,b,Aeq,beq,lb);
x =
    0.5000
    0.2500
    0.0000
    0.2500
fval =
   -0.1300
exitflag =
     1
```

可见，4 个项目的投资百分数分别为 0.50，0.25，0.00 和 0.25 时可使该单位获得最大的收益。最大收益为 13%。过程正常收敛。

【例 17-3】 工件加工任务分配问题。某车间有两台机床甲和乙，可用于加工 3 种工件。假定这两台机床的可用台时数分别为 700 和 800，3 种工件的数量分别为 300，500 和 400，且已知用两台不同机床加工单位数量的不同工件所需的台时数和加工费用（如表 17-2 所示）。问怎样分配机床的加工任务，才能既满足加工工件的要求，又使总加工费用最低。

表 17-2　机床加工情况表

机床类型	单位工作所需加工台时数			单位工件的加工费用			可用台时数
	工件 1	工件 2	工件 3	工件 1	工件 2	工件 3	
甲	0.4	1.1	1.0	13	9	10	700
乙	0.5	1.2	1.3	11	12	8	800

设在甲机床上加工工件 1，2 和 3 的数量分别为 x_1，x_2 和 x_3，在乙机床上加工工件 1，2 和 3 的数量分别为 x_4，x_5 和 x_6。根据 3 种工件的数量限制，有

$$x_1+x_4=300 \quad (\text{对工件 }1)$$

$$x_2+x_5=500 \quad (\text{对工件 }2)$$

$$x_3+x_6=400 \quad (\text{对工件 }3)$$

再根据机床甲和乙的可用总台时限制，可以得到其他约束条件。以总加工费用最少为目标函数，组合约束条件，可以得到下面的数学模型：

$$\begin{cases} \min z = 13x_1 + 9x_2 + 10x_3 + 11x_4 + 12x_5 + 8x_6 \\ x_1 + x_4 = 300 \\ x_2 + x_5 = 500 \\ x_3 + x_6 = 400 \\ 0.4x_1 + 1.1x_2 + x_3 \leqslant 700 \\ 0.5x_4 + 1.2x_5 + 1.3x_6 \leqslant 800 \\ x_j \geqslant 0, j = 1, 2, ..., 6 \end{cases}$$

首先输入下列系数：

```
f = [13;9;10;11;12;8];
A =  [0.4  1.1  1  0  0  0
      0  0  0  0.5  1.2  1.3];
b = [700; 800];
Aeq=[1 0 0 1 0 0
     0 1 0 0 1 0
     0 0 1 0 0 1];
beq=[300 500 400];
lb = zeros(6,1);
```

然后调用 linprog 函数：

```
[x,fval,exitflag,output,lambda] = linprog(f,A,b,Aeq,beq,lb);
x =
    0.0000
  500.0000
    0.0000
  300.0000
    0.0000
  400.0000
fval =
  1.1000e+004
exitflag =
     1
```

可见，在甲机床上加工 500 个工件 2，在乙机床上加工 300 个工件 1、加工 400 个工件 3 可在满足条件的情况下使总加工费最小。最小费用为 11000 元。收敛正常。

【例 17-4】 裁料问题。在某建筑工程施工中需要制作 10000 套钢筋，每套钢筋由 2.9m，2.1m 和 1.5m 3 种不同长度的钢筋各一根组成，它们的直径和材质不同。目前在市场上采购到的同类钢筋的长度每根均为 7.4m，问应购进多少根 7.4m 长的钢筋才能满足工程的需要？

首先分析共有多少种不同的套裁方法，该问题的可能材料方案如表 17-3 所示。

表 17-3　材料方案表

下料长度(m)	裁料方案编号							
	1	2	3	4	5	6	7	8
2.9	2	1	1	1	0	0	0	0
2.1	0	2	1	0	3	2	1	0
1.5	1	0	1	3	0	2	3	4
料头长度(m)	0.1	0.3	0.9	0	1.1	0.2	0.8	1.4

设以 $\boldsymbol{x_i}(i=1,2,\ldots,8)$表示按第 i 种裁料方案下料的原材料数量，则可得该问题的数学模型为

$$\begin{cases} \min z = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 \\ 2x_1 + x_2 + x_3 + x_4 = 10000 \\ 2x_2 + x_3 + 3x_5 + 2x_6 + x_7 = 10000 \\ x_1 + x_3 + 3x_4 + 2x_6 + 3x_7 + 4x_8 = 10000 \\ x_j \geqslant 0,\ j = 1, 2, ..., 8 \end{cases}$$

首先输入下列系数：

```
f = [1; 1; 1; 1; 1; 1; 1; 1];
Aeq=[2  0  0  0  0  0  0  0
      0  2  1  0  3  2  1  0
      1  0  1  3  0  2  3  4];
beq=[10000  10000  10000];
lb = zeros(8, 1);
```

然后调用 linprog 函数：

```
[x,fval,exitflag,output,lambda] = linprog(f,[],[],Aeq,beq,lb);
x =
  1.0e+003 *
    5.0000
    0.0000
    0.0000
    0.0000
    1.6667
    2.5000
    0.0000
    0.0000
fval =
  9.1667e+003
```

所以最节省的情况需要 9167 根 7.4m 长的钢筋，其中第 1 种方案使用 5000 根，第 5 种方案使用 1667 根，第 6 种方案使用 2500 根。

【例 17-5】　工作人员计划安排问题。某昼夜服务的公共交通系统每天各时间段（每 4 小时为一个时间段）所需的值班人数如表 17-4 所示。这些值班人员在某一时段开始上班后要连续工作 8 个小时（包括轮流用膳时间）。问该公交系统至少需要多少名工作人员才能满足值班的需要？

表 17-4　各时段所需值班人数表

班　　次	时 间 段	所 需 人 数
1	6:00～10:00	60
2	10:00～14:00	70
3	14:00～18:00	60
4	18:00～22:00	50
5	22:00～2:00	20
6	2:00～6:00	30

设 x_i 为第 i 个时段开始上班的人员数，据题意建立下面的数学模型：

$$\begin{cases} \min z = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \\ x_6 + x_1 \geqslant 60 \\ x_1 + x_2 \geqslant 70 \\ x_2 + x_3 \geqslant 60 \\ x_3 + x_4 \geqslant 50 \\ x_4 + x_5 \geqslant 20 \\ x_5 + x_6 \geqslant 30 \\ x_j \geqslant 0, j = 1, 2, ..., 6 \end{cases}$$

需要对前面 6 个约束条件进行形式变换，使不等式变为非正不等式。只需要在不等式两侧取负即可。

首先输入下列系数：

```
f = [1; 1; 1; 1; 1; 1];
A=[-1   0   0   0   0  -1
   -1  -1   0   0   0   0
    0  -1  -1   0   0   0
    0   0  -1  -1   0   0
    0   0   0  -1  -1   0
    0   0   0   0  –1  -1];
b=[-60; -70; -60; -50; -20; -30];
lb = zeros(6, 1);
```

然后调用 linprog 函数：

```
[x,fval,exitflag,output,lambda] = linprog(f,A,b,[ ],[ ],lb);
x =
   41.9176
   28.0824
   35.0494
   14.9506
    9.8606
   20.1394
fval =
  150.0000
exitflag =
     1
```

可见，只要 6 个时段分别安排 42 人、28 人、35 人、15 人、10 人和 20 人就可以满足值班的需要。共计 150 人。计算收敛。

【例 17-6】 厂址选择问题。考虑 A，B，C 三地，每地都出产一定数量的原料，也消耗一定数量的产品（见表 17-5）。已知制成每吨产品需 3 吨原料，各地之间的距离为：A～B：150km，A～C：100km，B～C：200km。假定每万吨原料运输 1km 的运价是 5000 元，每万吨产品运输 1km 的运价是 6000 元。由于地区条件的差异，在不同地点设厂的生产费用也不同。问究竟在哪些地方设厂，规模多大，才能使总费用最小？另外，由于其他条件限制，在 B 处建厂的规模（生产的产品数量）不能超过 5 万吨。

表 17-5　A，B，C 三地出产原料、消耗产品情况表

地　　点	年产原料（万吨）	年销产品（万吨）	生产费用（万元/万吨）
A	20	7	150
B	16	13	120
C	24	0	100

令 x_{ij} 为由 i 地运到 j 地的原料数量（万吨），y_{ij} 为由 i 地运往 j 地的产品数量（万吨），i=1, 2, 3, j=1, 2, 3（分别对应 A, B, C 三地）。根据题意，可以建立问题的数学模型（其中目标函数包括原材料运输费、产品运输费和生产费）：

$$
\begin{cases}
\min z = 75x_{12} + 75x_{21} + 50x_{13} + 50x_{31} + 100x_{23} + 100x_{32} \\
+150y_{11} + 240y_{12} + 210y_{21} + 120y_{22} + 160y_{31} + 220y_{32} \\
3y_{11} + 3y_{12} + x_{12} + x_{13} - x_{21} - x_{31} \leqslant 20 \\
3y_{21} + 3y_{22} - x_{12} + x_{21} + x_{23} - x_{32} \leqslant 16 \\
3y_{31} + 3y_{32} - x_{13} - x_{23} + x_{31} + x_{32} \leqslant 24 \\
y_{11} + y_{21} + y_{31} = 7 \\
y_{12} + y_{22} + y_{32} = 13 \\
y_{21} + y_{22} \leqslant 5 \\
x_{ij} \geqslant 0, i, j = 1,2,3; i \neq j \\
y_{ij} \geqslant 0, i = 1,2,3; j = 1, 2
\end{cases}
$$

首先输入下列系数：

```
f = [75;75;50;50;100;100;150;240;210;120;160;220];
A=[1 -1  1  -1  0  0  3  3  0  0  0  0
  -1  1  0  0  1 -1  0  0  3  3  0  0
   0  0 -1  1 -1  1  0  0  0  0  3  3
   0  0  0  0  0  0  0  0  1  1  0  0];
b=[20;16;24;5];
Aeq=[0 0 0 0 0 0 1 0 1 0 1 0
     0 0 0 0 0 0 0 1 0 1 0 1];
beq=[7;13];
lb = zeros(12,1);
```

然后调用 linprog 函数：

```
[x,fval,exitflag,output,lambda] = linprog(f,A,b,Aeq,beq,lb);
x =
   0.0000
```

```
      1.0000
      0.0000
      0.0000
      0.0000
      0.0000
      7.0000
      0.0000
      0.0000
      5.0000
      0.0000
      8.0000
fval =
      3.4850e+003
exitflag =
      1
```

要使总费用最小，需要 B 地向 A 地运送 1 万吨；A、B、C 三地的建厂规模分别为 7 万吨、5 万吨和 8 万吨。最小总费用为 3485 元。

17.2 有约束非线性最优化问题

17.2.1 基本数学原理

在有约束最优化问题中，通常要将该问题转换为更简单的子问题，这些子问题可以求解并作为迭代过程的基础。早期的方法通常是通过构造惩罚函数等来将有约束的最优化问题转换为无约束最优化问题进行求解。现在，这些方法已经被更有效的基于 K-T（Kuhn-Tucker）方程解的方法所取代。K-T 方程是有约束最优化问题求解的必要条件。假设有所谓的 Convex 规划问题，$f(\boldsymbol{x})$和 $\boldsymbol{G}_i(\boldsymbol{x})$，$i$=1,2,…,$m$ 为 Convex 函数，则 K-T 方程对于求得全局极小点是必要的，也是充分的。

对于规划问题

$$\min_{x\in R^n} f(\boldsymbol{x})$$

$$\boldsymbol{G_i}(\boldsymbol{x})=0 \qquad i=1,...,m_e$$

$$\boldsymbol{G_i}(\boldsymbol{x})\leqslant 0 \qquad i=m_e+1,...,m$$

$$\boldsymbol{x}_1\leqslant \boldsymbol{x}\leqslant \boldsymbol{x_u}$$

式中，$\boldsymbol{x}$ 是设计参数向量，（$\boldsymbol{x}\in \boldsymbol{R}^n$），$f(\boldsymbol{x})$ 为目标函数，返回标量值，向量函数 $\boldsymbol{G}(\boldsymbol{x})$ 返回等式约束和不等式约束在 x 处的值。

它的 K-T 方程可表达为

$$f(\boldsymbol{x}^*)+\sum_{i=1}^{m}\boldsymbol{\lambda}_i^*\cdot\nabla\boldsymbol{G}_i(\boldsymbol{x}^*)=0 \qquad (*)$$

$$\nabla\boldsymbol{G}_i(\boldsymbol{x}^*)=0 \qquad i=1,\ldots,m$$

$$\boldsymbol{\lambda}_i^*\geqslant 0 \qquad i=m_e+1,\ldots,m$$

其中第 1 行描述了目标函数和约束条件在解处梯度消失。由于梯度消失，需要用拉格朗日乘子λ_i (i=1,2,…,m) 来平衡目标函数与约束梯度间大小的差异。

K-T 方程的解形成了许多非线性规划算法的基础。这些算法直接计算拉格朗日乘子。用拟

牛顿法更新过程，给 K-T 方程积累二阶信息，可以保证有约束拟牛顿法的超线性收敛。这些方法称为序列二次规划法（SQP），因为在每一次主要的迭代中都求解一次二次规划问题。

对于给定的规划问题，序列二次规划（SQP）的主要思路是形成基于拉格朗日函数二次近似的二次规划子问题，即

$$L(\boldsymbol{x},\boldsymbol{\lambda}) = f(\boldsymbol{x}) + \sum_{i=1}^{m} \boldsymbol{\lambda}_i g_i(\boldsymbol{x})$$

这里，通过假设约束条件为不等式约束来使（*）式得到了简化，通过非线性有约束问题线性化来获得二次规划子问题。

二次规划子问题可表达为

$$\min_{d \in R^n} \frac{1}{2} \boldsymbol{d}^T \boldsymbol{H}_c d + \nabla f(\boldsymbol{x}_k)^T \boldsymbol{d}$$

$$\nabla g_i(\boldsymbol{x}_k)^T \boldsymbol{d} + g_i(\boldsymbol{x}_k) = 0 \qquad i=1,\ldots,m_e$$

$$\nabla g_i(\boldsymbol{x}_k)^T \boldsymbol{d} + g_i(\boldsymbol{x}_k) \leqslant 0 \qquad i= m_e+1,\ldots,m$$

该子问题可以用任意一种二次规划算法求解，求得的解可以用来形成新的迭代公式

$$x_k+1=x_k+\alpha\, kd_k$$

用 SQP 法求解非线性有约束问题时的迭代次数常比用解无约束问题时的少，因为在搜索区域内，SQP 方法可以获得最佳的搜索方向和步长信息。

给 Rosenbrock 函数添加非线性不等式约束 $g(x)$

$$x_1^2 + x_2^2 - 1.5 \leqslant 0$$

经过 96 次迭代得到问题的解：$\boldsymbol{x}$=[0.9072,0.8288]，初值为 $\boldsymbol{x}$=[-1.9, 2]，无约束问题则需要 140 次迭代。图 17-1 是搜索路径图。

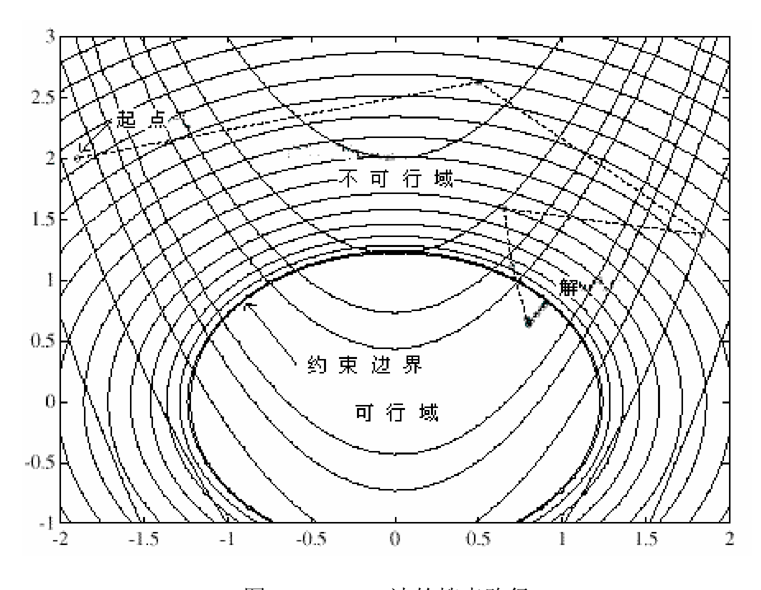


图 17-1 SQP 法的搜索路径

MATLAB 中 SQP 法的实现分 3 步，即

① 拉格朗日函数 Hess 矩阵的更新；

② 二次规划问题求解；

③ 一维搜索和目标函数的计算。

1．Hess 矩阵的更新

在每一次主要迭代过程中，都用 BFGS 法计算拉格朗日函数的 Hess 矩阵的拟牛顿近似矩阵。更新公式为

$$\boldsymbol{H}_{k+1}=\boldsymbol{H}_k+\frac{\boldsymbol{q}_k\boldsymbol{q}_k^T}{\boldsymbol{q}_k^T\boldsymbol{S}_k}-\frac{\boldsymbol{H}_k^T\boldsymbol{H}_k}{\boldsymbol{S}_k^T\boldsymbol{H}_k\boldsymbol{S}_k}$$

式中

$$\boldsymbol{S}_k=x_{k+1}-x_k$$

$$\boldsymbol{q}_k=\nabla f(x_{k+1})+\sum_{i=1}^{n}\lambda_i\cdot\nabla g_i(x_{k+1})-(\nabla f(x_k)+\sum_{i=1}^{n}\lambda_i-\nabla g_i(x_k))$$

上式中，$\lambda_i(i=1,2,\ldots,m)$为对拉格朗日乘子的估计。

2．二次规划求解

SQP 法的每一次主要迭代过程都要求一次二次规划问题，形式如下：

$$\min_{d\in R^n} q(\boldsymbol{d})=\frac{1}{2}\boldsymbol{d}^T\boldsymbol{H}\boldsymbol{d}+\boldsymbol{c}^T\boldsymbol{d}$$

$$\boldsymbol{A}_i\boldsymbol{d}=\boldsymbol{b}_i \qquad i=1,\ldots,m_e$$

$$\boldsymbol{A}_i\boldsymbol{d}\leqslant\boldsymbol{b}_i \qquad i=m_e+1,\ldots,m$$

求解过程分两步，第 1 步涉及可行点（若存在）的计算，第 2 步为可行点至解的迭代序列。在第 1 步中，需要有可行点作为初值，若当前点不可行，则通过求解下列线性规划问题可以得到一个可行点：

$$\min_{\gamma\in R,x\in R^n}\gamma$$

$$\boldsymbol{A}_i\boldsymbol{X}=\boldsymbol{b}_i \qquad i=1,\ldots,m_e$$

$$\boldsymbol{A}_i\boldsymbol{X}-\gamma\leqslant\boldsymbol{b}_i \qquad i=m_e+1,\ldots,m$$

式中，$\boldsymbol{A}_i$为矩阵 $\boldsymbol{A}$ 的第 i 行。

3．一维搜索和目标函数的计算

二次规划子问题的解生成一个向量 $\boldsymbol{d}_k$，它形成一个新的迭代公式：

$$\boldsymbol{X}_{k+1}=\boldsymbol{X}_k+\alpha\boldsymbol{d}_k$$

α为步长参数。

目标函数的形式如下：

$$\psi(\boldsymbol{x})=f(\boldsymbol{x})+\sum_{i=1}^{m_e}\gamma_i\cdot g_i(\boldsymbol{x})+\sum_{i=m_e+1}^{m}\gamma_i\cdot\max\{0,g_i(\boldsymbol{x})\}$$

式中

$$\gamma_i=(\gamma_{k+1})_i=\max_i\{\lambda_i\cdot\frac{1}{2}((\lambda_k)_i+\lambda_i)\} \qquad i=1,\ldots,m$$

17.2.2　有关函数介绍

利用 fmincon 函数求多变量有约束非线性函数的最小值。假设多变量非线性函数的数学模型为：

$$\min_x f(\boldsymbol{x})$$
$$\boldsymbol{c}(\boldsymbol{x}) \leqslant 0$$
$$\boldsymbol{ceq}(\boldsymbol{x}) = 0$$
$$\boldsymbol{A} \cdot \boldsymbol{x} \leqslant \boldsymbol{b}$$
$$\boldsymbol{Aeq} \cdot \boldsymbol{x} \leqslant \boldsymbol{beq}$$
$$\boldsymbol{lb} \leqslant \boldsymbol{x} \leqslant \boldsymbol{ub}$$

式中，$\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{beq}, \boldsymbol{lb}$,和 $\boldsymbol{ub}$ 为向量，$\boldsymbol{A}$ 和 $\boldsymbol{Aeq}$ 为矩阵，$\boldsymbol{c}(\boldsymbol{x})$和 $\boldsymbol{ceq}(\boldsymbol{x})$为函数，返回标量。$f(\boldsymbol{x}), \boldsymbol{c}(\boldsymbol{x})$, 和 $\boldsymbol{ceq}(\boldsymbol{x})$可以是非线性函数。

fmincon 函数的调用格式如下：

- fmincon 求多变量有约束非线性函数的最小值，常用于有约束非线性优化问题。
- x = fmincon(fun,x0,A,b) 给定初值 x0，求解 fun 函数的最小值 x。fun 函数的约束条件为 A*x <= b，x0 可以是标量、向量或矩阵。
- x = fmincon(fun,x0,A,b,Aeq,beq) 最小化 fun 函数，约束条件为 Aeq*x = beq 和 A*x <= b。若没有不等式存在，则设置 A=[ ]，b=[ ]。
- x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub) 定义设计变量 x 的下界 lb 和上界 ub，使得总是有 lb <= x <= ub。若无等式存在，则令 Aeq=[ ]，beq=[ ]。
- x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon) 在上面的基础上，在 nonlcon 参数中提供非线性不等式 $c(x)$或等式 $ceq(x)$。 fmincon 函数要求 c(x) <= 0 且 ceq(x) = 0。当无边界存在时，令 lb=[ ]和（或）ub=[ ]。
- x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options) 用 optiions 参数指定的参数进行最小化。
- x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options,P1,P2,...) 将问题参数 P1, P2 等直接传递给函数 fun 和 nonlin。若不需要这些变量，则传递空矩阵到 A, b, Aeq, beq, lb, ub, nonlcon 和 options。
- [x,fval] = fmincon(...) 返回解 x 处的目标函数值。
- [x,fval,exitflag] = fmincon(...) 返回 exitflag 参数，描述函数计算的退出条件。
- [x,fval,exitflag,output] = fmincon(...) 返回包含优化信息的输出参数 output。
- [x,fval,exitflag,output,lambda] = fmincon(...) 返回解x处包含拉格朗日乘子的 lambda 参数。
- [x,fval,exitflag,output,lambda,grad] = fmincon(...) 返回解x处fun 函数的梯度。
- [x,fval,exitflag,output,lambda,grad,hessian] = fmincon(...) 返回解x处fun 函数的 Hess 矩阵。

各调用格式中，nonlcon 参数计算非线性不等式约束 $c(x)\leqslant 0$ 和非线性等式约束 $ceq(x)=0$。nonlcon 参数是一个包含函数名的字符串。该函数可以是 M 文件、内部文件或 MEX 文件。它要求输入一个向量 x，返回两个变量——解 x 处的非线性不等式向量 $\boldsymbol{c}$ 和非线性等式向量 $\boldsymbol{ceq}$。例如，若 nonlcon='mycon'，则 M 文件 mycon.m 具有下面的形式：

```
function [c,ceq] = mycon(x)
c = ...     % 计算 x 处的非线性不等式。
ceq = ...   % 计算 x 处的非线性等式。
```

若还计算了约束的梯度，即

```
options = optimset('GradConstr','on')
```

则 nonlcon 函数必须在第 3 个和第 4 个输出变量中返回 $c(x)$的梯度 GC 和 $ceq(x)$的梯度 Gceq。

当被调用的 nonlcon 函数只需要两个输出变量（此时优化算法只需要 c 和 ceq 的值，而不需要 GC 和 GCeq）时，可以通过查看 nargout 的值来避免计算 GC 和 Gceq 的值。

```
function [c,ceq,GC,GCeq] = mycon(x)
c = ...             % 解 x 处的非线性不等式。
ceq = ...           % 解 x 处的非线性等式。
if nargout > 2      % 被调用的 nonlcon 函数，要求有 4 个输出变量。
   GC = ...         % 不等式的梯度。
   GCeq = ...       % 等式的梯度。
end
```

若 nonlcon 函数返回 m 元素的向量 $\boldsymbol{c}$ 和长度为 n 的 $\boldsymbol{x}$，则 $\boldsymbol{c}(\boldsymbol{x})$的梯度 $\boldsymbol{GC}$ 是一个 $n \times m$ 的矩阵，其中 $\boldsymbol{GC}(i, j)$是 $\boldsymbol{c}(j)$对 $\boldsymbol{x}(i)$的偏导数。同样，若 $\boldsymbol{ceq}$ 是一个 p 元素的向量，则 $\boldsymbol{ceq}(\boldsymbol{x})$的梯度 $\boldsymbol{Gceq}$ 是一个 $n \times p$ 的矩阵，其中 $\boldsymbol{Gceq}(i, j)$是 $\boldsymbol{ceq}(j)$对 $\boldsymbol{x}(i)$的偏导数。

其他参数意义可以参见表 15-7 和表 15-8。

根据问题规模的不同，fmincon 函数使用不同的优化算法：

- 大型优化算法：默认时，若提供了函数的梯度信息，并且只有上下界存在或只有线性等式约束存在，则 fmincon 函数将选择大型算法。本法是基于内部映射牛顿法(interior-reflective Newton method)的子空间置信域法(subspace trust-region)。该法的每一次迭代都与用 PCG 法求解大型线性系统得到的近似解有关。
- 中型优化算法：fmincon 函数使用序列二次规划法(SQP)。本法中，在每一步迭代中求解二次规划子问题，并用 BFGS 法更新拉格朗日 Hess 矩阵。

在使用该函数的过程中，还有一些需要注意的问题：

（1）大型优化问题

① 使用大型算法，必须在 fun 函数中提供梯度信息(options.GradObj 设置为'on')。如果没有梯度信息，则给出警告消息。

fmincon 函数允许 g(x)为一近似梯度，但使用真正的梯度将使优化过程更具稳健性。

② 当对矩阵的二阶导数（即 Hess 矩阵）进行计算以后，用该函数求解大型问题将更有效。但不需要求得真正的 Hess 矩阵，如果能提供 Hess 矩阵的稀疏结构的信息（用 options 参数的 HessPattern 属性），则 fmincon 函数可以算得 Hess 矩阵的稀疏有限差分近似。

③ 若 x0 不是严格可行的，则 fmincon 函数选择一个新的严格可行初始点。

④ 若 x 的某些元素没有上界或下界，则 fmincon 函数更希望对应的元素设置为 Inf（对于上界）或-Inf（对于下界），而不希望强制性地给上界赋一个很大的值，给下界赋一个很小的负值。

⑤ 线性约束最小化课题中也有几个问题需要注意：

- Aeq 矩阵中若存在密集列或近密集列（A dense (或 fairly dense)column），将导致满秩并使计算费时。
- fmincon 函数剔除 Aeq 中线性相关的行。此过程需要进行反复的因子分解，因此，如果相关行很多的话，计算将是一件很费时的事情。
- 每一次迭代都要用下式进行稀疏最小二乘求解

$$\boldsymbol{B} = \boldsymbol{Aeq}^{T} \boldsymbol{R}^{-T}$$

式中 $\boldsymbol{R}^{\boldsymbol{T}}$ 为前提条件的乔累斯基因子。

（2）中型优化问题

① 如果用 *Aeq* 和 *beq* 清楚地提供等式约束，将比用 *lb* 和 *ub* 获得更好的数值解。

② 在二次子问题中，若有等式约束并且诸因变等式（dependent equalities）被发现和剔除，则将在过程标题中显示'dependent'(当 output 参数要求使用 options.Display = 'iter')。只有在等式连续的情况下，因变等式才会被剔除。若等式系统不连续，则子问题将不可行并在过程标题中输出'infeasible'消息。

求大型优化问题的代码中不允许上限和下限相等，即不能有 ***lb***(2)==***ub***(2),否则给出下面的出错消息：

```
Equal upper and lower bounds not permitted in this large-scale method.
Use equality constraints and the medium-scale method instead.
```

若只有等式约束，仍然可以使用大型算法。当既有等式约束又有边界约束时，使用中型算法。

（3）使用 fmincon 函数的一些要求

① 目标函数和约束函数都必须是连续的，否则可能会给出局部最优解。

② 当问题不可行时，fmincon 函数将试图使最大约束值最小化。

③ 目标函数和约束函数都必须是实数。

④ 对于大型优化问题，使用大型优化算法时，用户必须在 fun 函数中提供梯度(options 参数的 GradObj 属性必须设置为'on')，并且只可以指定上界和下界约束，或者只有线性约束必须存在，Aeq 的行数不能多于列数。

⑤ 现在，如果在 fun 函数中提供了解析梯度，则选项参数 DerivativeCheck 不能与大型方法一起用，以比较解析梯度和有限差分梯度。可以通过将 options 参数的 MaxIter 属性设置为 0 来用中型方法核对导数。然后用大型方法求解问题。

17.2.3　应用实例

【例 17-6】 求侧面积为常数 $6\times5^2\mathrm{m}^2$ 的体积最大的长方体体积。设该长方体的长、宽、高分别为 x_1、x_2 和 x_3，据题意得到下面的数学模型：

$$\begin{cases} \min z = -x_1x_2x_3 \\ 2(x_2x_3+x_3x_1+x_1x_2)=150 \end{cases}$$

编一个 M 文件 optim17_6o.m，返回 x 处的函数值 f。

```
function f = myfun(x)
f = -x(1) * x(2) * x(3);
```

由于约束条件是非线性等式约束，所以需要编写一个约束条件 M 文件 opt17_6c.m：

```
function [c,ceq] = mycon(x)
ceq =x(2)x(3)+x(3)x(1)+x(1)x(2) -75
```

下一步给定初值，并调用优化过程。

```
x0 = [4; 5; 6];
lb=zeros(3,1);
[x,fval,exitflag,output,lambda]=fmincon(@opt17_6o,x0,[],
[],[],[], lb,[],@opt17_6c)
```

计算结果为：

```
Optimization terminated successfully:
 Search direction less than 2*options.TolX and
```

```
  maximum constraint violation is less than options.TolCon
Active Constraints:
     1
x =
    5.0000
    5.0000
    5.0000
fval =
 -125.0000
exitflag =
     1
output =
       iterations: 7
        funcCount: 38
         stepsize: 1
        algorithm: 'medium-scale: SQP, Quasi-Newton, line-search'
    firstorderopt: []
     cgiterations: []
lambda =
         lower: [3x1 double]
         upper: [3x1 double]
         eqlin: [0x1 double]
      eqnonlin: 2.5000
       ineqlin: [0x1 double]
    ineqnonlin: [0x1 double]
```

优化结果显示过程成功收敛，搜索方向小于两倍 options.TolX，最大违约值小于options.TolCon，主动约束为 1 个。

问题的解为 $x(1)=x(2)=x(3)=5.0000$m，最大体积为 125.0000m³。exitflag=1，表示过程成功收敛于解 $\boldsymbol{x}$ 处。output 输出变量显示了收敛过程中的迭代次数、目标函数计算次数、步长、算法等信息。lambda 则包含模型信息。

【例 17-7】 试设计一压缩圆柱螺旋弹簧，要求其质量最小。弹簧材料为 65Mn，最大工作载荷 P_{max}=40N，最小工作载荷为 0，载荷变化频率 f_r= 25Hz，弹簧寿命为 104h，弹簧钢丝直径 d 的取值范围为 1～4mm，中径 D_2 的取值范围为 10～30mm，工作圈数 n 不应小于 4.5 圈，弹簧旋绕比 C 不应小于 4，弹簧一端固定，一端自由，工作温度为 50℃，弹簧变形量不小于 10mm。

本题的优化目标是使弹簧质量最小，圆柱螺旋弹簧的质量可以表示为

$$M \approx \gamma(n+n_2)\pi D_2 \frac{\pi}{4} d^2$$

式中：γ ——弹簧材料的密度，对于钢材 γ =7.8×10⁻⁶kg/mm³；

n——工作圈数；

n_2——死圈数，常取 n_2=1.5～2.5，现取 n_2=2；

D_2——弹簧中径，（mm）；

d——弹簧钢丝直径，（mm）。

将已知参数代入公式，进行整理以后得到问题的目标函数为

$$f(\boldsymbol{X}) = M = 0.192457\times10^{-4}(x_2+2)x_1^2 x_3$$

根据弹簧性能和结构上的要求，可写出问题的约束条件：

（1）强度条件

$$g_1(\boldsymbol{X}) = 350 - 163.0x_1^{-2.86}x_3^{0.86} \geqslant 0$$

（2）刚度条件

$$g_2(\boldsymbol{X}) = 0.4\times10^{-2}x_1^{-4}x_2x_3^3 - 10.0 \geqslant 0$$

（3）稳定性条件

$$g_3(\boldsymbol{X}) = 3.7x_3 - (x_2+1.5)x_1 - 0.44\times10^{-2}x_1^{-4}x_2x_3^3 \geqslant 0$$

（4）不发生共振现象，要求

$$g_4(\boldsymbol{X}) = 0.356\times10^6 x_1x_2^{-1}x_3^{-2} - 375 \geqslant 0$$

（5）弹簧旋绕比的限制

$$g_5(\boldsymbol{X}) = x_3x_1^{-1} - 4.0 \geqslant 0$$

（6）对 d, n, D_2 的限制

$$1.0 \leqslant d \leqslant 4.0$$

且应取标准值，即 1.0, 1.2, 1.6, 2.0, 2.5, 3.0, 3.5, 4.0mm 等。

$$4.5 \leqslant n \leqslant 50$$

$$10 \leqslant D_2 \leqslant 30$$

由上可知，该压缩圆柱螺旋弹簧的优化设计是一个三维的约束优化问题，其数学模型为

$$\min f(\boldsymbol{X}) = M = 0.192457\times10^{-4}(x_2+2)x_1^2x_3$$

$$\begin{cases} g_1(\boldsymbol{X}) = 350 - 163x_1^{-2.86}x_3^{0.86} \leqslant 0 \\ g_2(\boldsymbol{X}) = 0.4\times10^{2}x_1^{-4}x_2x_3^3 - 10 \geqslant 0 \\ g_3(\boldsymbol{X}) = 3.7x_3 - (x_2+1.5)x_1 - 0.44\times10^{-2}x_1^{-4}x_2x_3^3 \geqslant 0 \\ g_4(\boldsymbol{X}) = 0.356\times10^6x_1x_2^{-1}x_3^{-2} - 375 \geqslant 0 \\ g_5(\boldsymbol{X}) = x_3/x_1 - 4 \geqslant 0 \\ g_6(\boldsymbol{X}) = x_1 - 1 \geqslant 0 \\ g_7(\boldsymbol{X}) = 4 - x_1 \geqslant 0 \\ g_8(\boldsymbol{X}) = x_2 - 4.5 \geqslant 0 \\ g_9(\boldsymbol{X}) = 50 - x_2 \geqslant 0 \\ g_{10}(\boldsymbol{X}) = x_3 - 10 \geqslant 0 \\ g_{11}(\boldsymbol{X}) = 30 - x_3 \geqslant 0 \end{cases}$$

取初始设计参数为 $\boldsymbol{X}^{(0)}$=[2.0, 5.0, 25.0]^{T}

首先编写目标函数的 M 文件 opt17_7.m，返回 x 处的函数值 f。

```
function f = myfun(x)
f = 0.192457*1e-4*( x(2)+2)*x(1)^2 * x(3);
```

由于约束条件中有非线性约束，所以需要编写一个描述非线性约束条件的 M 文件 opt17_7c.m：

```
function [c,ceq] = mycon(x)
c(1)=350-163*x(1)^( -2.86)*x(3)^0.86;
c(2)=10-0.4*0.01*x(1)^( -4)*x(2)*x(3)^3;
c(3)=(x(2)+1.5)*x(1)+0.44*0.01*x(1)^( -4)*x(2)*x(3)^3-3.7*x(3);
c(4)=375-0.356*1e6*x(1)*x(2)^( -1)*x(3)^( -2);
```

```
c(5)=4-x(3)/x(1);
```

然后设置线性约束的系数：

```
A=[-1    0   0
    1    0   0
    0   -1   0
    0    1   0
    0    0  -1
    0    0   1];
b=[-1;4;-4.5;50;-10;30];
```

下一步给定初值，给定变量的下限约束，并调用优化过程。

```
x0 = [2.0; 5.0; 25.0];
lb=zeros(3,1);
[x,fval,exitflag,output,lambda]=fmincon(@opt17_7o,x0,A,b,[],[],
lb,[],@opt17_7c)
```

计算结果为

```
x =
    1.6564
    4.5000
   16.1141
fval =
    0.0055
exitflag =
     1
output =
       iterations: 3
        funcCount: 16
         stepsize: 1
        algorithm: 'medium-scale: SQP, Quasi-Newton, line-search'
    firstorderopt: []
     cgiterations: []
```

所以当弹簧钢丝的直径 d、工作圈数 n 及中径 D_2 分别取 1.6564、4.5000 和 16.1141 时弹簧质量最小，为 5.5 克。考虑到实际情况，各参数可分别取 1.6，5.0 和 16.0

```
x=[1.6 5.0 16.0];
z=optim253(x)
z =
    0.0055
```

故此时弹簧质量仍为 5.5 克。

第 18 章　二 次 规 划

18.1　基本数学原理

如果某非线性规划的目标函数为自变量的二次函数，约束条件全是线性函数，就称这种规划为二次规划。其数学模型为：

$$\min_{x} \frac{1}{2} \boldsymbol{x}^T \boldsymbol{H}\boldsymbol{x} + \boldsymbol{f}^T \boldsymbol{x}$$
$$\boldsymbol{A} \cdot \boldsymbol{x} \leqslant \boldsymbol{b}$$
$$\boldsymbol{Aeq} \cdot \boldsymbol{x} \leqslant \boldsymbol{beq}$$
$$\boldsymbol{lb} \leqslant \boldsymbol{x} \leqslant \boldsymbol{ub}$$

式中，***H***, ***A***,和 ***Aeq*** 为矩阵, ***f***, ***b***, ***beq***, ***lb***, ***ub***,和 ***x*** 为向量。

18.2　有关函数介绍

（1）利用 quadprog 函数求解二次规划问题，其调用格式为：

● x = quadprog(H, f, A, b) 返回向量 x，使函数 1/2 * x' * H * x + f' * x 最小化，其约束条件为 A*x <= b。

● x = quadprog(H, f, A, b, Aeq, beq) 仍然求解上面的问题，但添加了等式约束条件 Aeq*x = beq。

● x = quadprog(H, f, A, b, lb, ub)定义设计变量的下界 lb 和上界 ub，使得 lb <= x <= ub。

● x = quadprog(H, f, A, b, lb, ub, x0)同上，并设置初值 x0。

● x = quadprog(H, f, A, b, lb, ub, x0, options)根据 options 参数指定的优化参数进行最小化。

● [x, fval] = quadprog(...)返回解 x 处的目标函数值 fval = 0.5*x'*H*x + f ' *x。

● [x, fval, exitflag] = quadprog(...)返回 exitflag 参数，描述计算的退出条件。

● [x, fval, exitflag, output] = quadprog(...)返回包含优化信息的结构输出 output。

● [x, fval, exitflag, output, lambda] = quadprog(...)返回解 x 处包含拉格朗日乘子的 lambda 参数。

各参数的意义可参见表 15-7 和表 15-8。

（2）根据问题的规模，quadprog 函数可使用不同的优化算法如下：

① 大型优化算法：当优化问题只有上界和下界，而没有线性不等式或等式约束，则默认

算法为大型算法。或者，如果优化问题中只有线性等式，而没有上界和下界或线性不等式时，默认算法也是大型算法。大型算法是基于内部映射牛顿法(interior-reflective Newton method)的子空间置信域法(subspace trust-region)。该法的每一次迭代都与用 PCG 法求解大型线性系统得到的近似解有关。

② 中型优化算法：quadprog 函数使用活动集法，它也是一种投影法，首先通过求解线性规划问题来获得初始可行解。

注意：

① 一般地，如果问题不是严格凸性的，用 quadprog 函数得到的可能是局部最优解。

② 如果用 ***Aeq*** 和 ***Beq*** 明确地指定等式约束，而不是用 ***lb*** 和 ***ub*** 指定，则可以得到更好的数值解。

③ 若 x 的组分没有上限或下限，则 quadprog 函数希望将对应的组分设置为 Inf（对于上限）或-Inf（对于下限），而不是强制性地给予上限一个很大的数或给予下限一个很小的负数。

④ 对于大型优化问题，若没有提供初值 ***x*0**，或 ***x*0** 不是严格可行，则 quadprog 函数会选择一个新的初始可行点。

⑤ 若为等式约束，且 quadprog 函数发现负曲率(negative curvature)，则优化过程终止，exitflag 的值等于-1。

（3）在使用 quadprog 函数的过程中，需要注意以下一些问题。

① 大型优化问题。大型优化问题不允许约束上限和下限相等，若 ***lb***(2)==***ub***(2)，则给出以下出错消息：

Equal upper and lower bounds not permitted in this large-scale method.Use equality constraints and the medium-scale method instead.

若优化模型中只有等式约束，仍然可以使用大型算法；如果模型中既有等式约束又有边界约束，则必须使用中型方法。

② 中型优化问题。当解不可行时，quadprog 函数给出以下警告：

Warning: The constraints are overly stringent;there is no feasible solution.

这里，quadprog 函数生成使约束矛盾最坏程度最小的结果。当等式约束不协调时，给出下面的警告消息：

Warning: The equality constraints are overly stringent;there is no feasible solution.

当 Hess 矩阵为负半定时，生成无边界解，给出下面的警告消息：

Warning: The solution is unbounded and at infinity;the constraints are not restrictive enough.

这里，quadprog 函数返回满足约束条件的 ***x*** 值。

另外，使用该函数时还有下面一些要求：

- 此时，显示水平只能选择 'off ' 和 'final' ，迭代参数 'iter' 不可用。
- 当问题不定或负定时，常常无解（此时 exitflag 参数给出一个负值，表示优化过程不收敛）。若正定解存在，则 quadprog 函数可能只给出局部极小值，因为问题可能是非凸的。
- 对于大型问题，不能依靠线性等式，因为 ***Aeq*** 必须是行满秩的，即 ***Aeq*** 的行数必须不多于列数。若不满足要求，则必须调用中型算法进行计算。

18.3　应用实例

【例 18-1】　求解下面的最优化问题：

目标函数　$f(\boldsymbol{x})=\frac{1}{2}x_1^2+x_2^2-x_1x_2-2x_1-6x_2$

约束条件

$$
\begin{aligned}
& x_1 + x_2 \leqslant 2 \\
& -x_1 + 2x_2 \leqslant 2 \\
& 2x_1 + x_2 \leqslant 3 \\
& 0 \leqslant x_1, 0 \leqslant x_2
\end{aligned}
$$

首先，目标函数可以写成下面的矩阵形式：

$$
H = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix},\quad f = \begin{bmatrix} -2 \\ -6 \end{bmatrix},\quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}
$$

输入下列系数矩阵：

```
H = [1 -1; -1 2]
f = [-2; -6]
A = [1 1; -1 2; 2 1]
b = [2; 2; 3]
lb = zeros(2,1)
```

然后调用二次规划函数 quadratic：

```
[x, fval, exitflag, output, lambda] = quadprog(H, f, A, b, [], [], lb)
```

得问题的解：

```
x =
    0.6667
    1.3333
fval =
   -8.2222
exitflag =
    1
output =
       iterations: 3
        algorithm: 'medium-scale: active-set'
   firstorderopt: []
    cgiterations: []
lambda.ineqlin
ans =
   3.1111
   0.4444
        0
lambda.lower
ans =
    0
    0
```

第 19 章　多目标规划

19.1　算法

前面介绍的最优化方法只有一个目标函数，是单目标最优化方法。但是，在许多实际工程问题中，往往希望多个指标都达到最优值，所以就有多个目标函数。这种问题称为多目标最优化问题。多目标最优化问题的数学模型为：

$$\min_{x\in R^n} \boldsymbol{F}(\boldsymbol{x})$$

$$\boldsymbol{G}_i(\boldsymbol{x})=0 \qquad i=1,\ldots,m_e$$

$$\boldsymbol{G}_i(\boldsymbol{x})\leqslant 0 \qquad i=m_e+1,\ldots,m$$

$$\boldsymbol{x}_i\leqslant \boldsymbol{x}\leqslant \boldsymbol{x}_u$$

式中 $\boldsymbol{F}(\boldsymbol{x})$为目标函数向量。

由于多目标最优化问题中各目标函数之间往往是不可公度的，因此往往没有惟一解，此时必须引进非劣解的概念（非劣解又称为有效解或帕累托解）。

定义：若 $x^*(x^*\in\Omega)$的邻域内不存在Δx，使得$(x^*+\Delta x)\in\Omega$，且

$$F_i(x^*+\Delta x)\leqslant F_i(x^*) \qquad i=1,\ldots,m$$

$$F_j(x^*+\Delta x)<F_j(x^*) \qquad \text{对于某些 } j$$

则称 x^*为非劣解

多目标规划有许多解法，下面列出常用的几种。

1．权和法

该法将多目标向量问题转化为所有目标的加权求和的标量问题，即

$$\min_{x\in\Omega} f(x)=\sum_{i=1}^{m}\omega_i\cdot F_i(x)^2$$

加权因子的选取方法很多，有专家打分法、α 方法、容限法和加权因子分解法等。

该问题可以用标准的无约束最优化算法进行求解。

2．ε 约束法

ε 约束法克服了权和法的某些凸性问题。它对目标函数向量中的主要目标 $\boldsymbol{F}_p$ 进行最小化，将其他目标用不等式约束的形式写出：

$$\min_{x\in\Omega} \boldsymbol{F}_p(\boldsymbol{x})$$

$$\text{sub.}\quad \boldsymbol{F}_i(\boldsymbol{x})\leqslant\varepsilon_i \qquad i=1,\ldots,m \quad i\neq p$$

3．目标达到法

目标函数系列为 $F(x)=\{F_1(x),\ F_2(x),\ldots,F_m(x)\}$，对应地有其目标值系列 $F^*=\{F_1^*,F_2^*,\ldots,F_m^*\}$。允许目标函数有正负偏差，偏差的大小由加权系数向量 $\boldsymbol{W}=\{W_1,W_2,\ldots,W_m\}$控制，于是目标达到问题可以表达为标准的最优化问题：

$$\min_{\gamma \in R, x \in \Omega} \gamma$$

$$\text{sub.} \quad \boldsymbol{F}_i(\boldsymbol{x}) - \boldsymbol{\omega}_i \gamma \leqslant \boldsymbol{F}_i^* \qquad i=1,\ldots,m$$

指定目标{ F_1^*, F_2^* }，定义目标点 P。权重向量定义从 P 到可行域空间 $\Lambda(\gamma)$ 的搜索方向。在优化过程中，γ 的变化改变可行域的大小，约束边界变为惟一解点 F_{1s}，F_{2s}。

4．目标达到法的改进

目标达到法的一个好处是可以将多目标最优化问题转化为非线性规划问题。但是，在序列二次规划（SQP）过程中，一维搜索的目标函数选择不是一件容易的事情，因为在很多情况下，很难决定是使目标函数变大好还是使它变小好。这导致许多目标函数创建过程的提出。可以通过将目标达到问题变为最大最小化问题来获得更合适的目标函数。

$$\min_{x \in R^n} \max_i \{\Lambda_i\}$$

式中

$$\Lambda_i = \frac{F_i(x) - F_i^*}{W_i} \qquad i = 1, \ldots, m$$

19.2 有关函数介绍

利用 fgoalattain 函数求解多目标达到问题。假设多目标达到问题的数学模型为

$$\begin{aligned} &\min_{x,\gamma} \gamma \\ &\boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{weight} \cdot \gamma \leqslant \boldsymbol{goal} \\ &\boldsymbol{c}(\boldsymbol{x}) \leqslant 0 \\ &\boldsymbol{ceq}(\boldsymbol{x}) = 0 \\ &\boldsymbol{A} \cdot \boldsymbol{x} \leqslant \boldsymbol{b} \\ &\boldsymbol{Aeq} \cdot \boldsymbol{x} = \boldsymbol{beq} \\ &\boldsymbol{lb} \leqslant \boldsymbol{x} \leqslant \boldsymbol{ub} \end{aligned}$$

式中 ***x***, ***weight***, ***goal***, ***b***, ***beq***, ***lb*** 和 ***ub*** 为向量, ***A*** 和 ***Aeq*** 为矩阵, ***c***(***x***), ***ceq***(***x***)和 ***F***(***x***)为函数，返回向量。***F***(***x***), ***c***(***x***)和 ***ceq***(***x***)可以是非线性函数。

fgoalattain 函数的调用格式为：

- fgoalattain 求解多目标达到问题。
- x = fgoalattain(fun, x0, goal, weight)试图通过变化 x 来使目标函数 fun 达到 goal 指定的目标。初值为 x0,weight 参数指定权重。
- x = fgoalattain(fun, x0, goal, weight, A, b)求解目标达到问题，约束条件为线性不等式 A*x <= b。
- x = fgoalattain(fun, x0, goal, weight, A, b, Aeq, beq)求解目标达到问题，除提供上面的线性不等式以外，还提供线性等式 Aeq*x = beq 。当没有不等式存在时，设置 A=[ ]，b=[ ]。
- x = fgoalattain(fun, x0, goal, weight, A, b, Aeq, beq, lb, ub)为设计变量 x 定义下界 lb 和上界 ub 集合，这样始终有 lb <= x <= ub。
- x = fgoalattain(fun, x0, goal, weight, A, b, Aeq, beq, lb, ub, nonlcon) 将目标达到问题归结为 nonlcon 参数定义的非线性不等式 *c*(*x*)或非线性等式 *ceq*(*x*)。fgoalattain 函数优化的约束条件

为(x) <= 0 和 ceq(x) = 0。若不存在边界，则设置 lb=[ ]和（或）ub=[ ]。

- x = fgoalattain(fun, x0, goal, weight, A, b, Aeq, beq, lb, ub, nonlcon, ... options)用 options 中设置的优化参数进行最小化。
- x = fgoalattain(fun, x0, goal, weight, A, b, Aeq, beq, lb, ub, nonlcon, ... options, P1, P2, ...)将问题参数 P1, P2 等直接传递给函数 fun 和 nonlcon。如果不需要参数 A, b, Aeq, beq, lb, ub, nonlcon 和 options，则将它们设置为空矩阵。
- [x, fval] = fgoalattain(...) 返回解 x 处的目标函数值。
- [x, fval, attainfactor] = fgoalattain(...) 返回解 x 处的目标达到因子。
- [x, fval, attainfactor, exitflag] = fgoalattain(...) 返回 exitflag 参数，描述计算的退出条件。
- [x, fval, attainfactor, exitflag, output] = fgoalattain(...)返回包含优化信息的输出参数 output。
- [x, fval, attainfactor, exitflag, output, lambda] = fgoalattain(...) 返回包含拉格朗日乘子的 lambda 参数。

各调用格式中，goal 变量为目标希望达到的向量值。向量的长度与 fun 函数返回的目标数 F 相等。fgoalattain 函数试图通过使向量 F 的值最小化来达到 goal 参数给定的目标。

nonlcon 参数计算非线性不等式约束 $c(x)$ <=0 和非线性等式约束 $ceq(x)$=0。nonlcon 函数是一个包含函数名的字符串，该函数可以是 M 文件、内部函数或 MEX 文件。nonlcon 函数需要输入向量 $\boldsymbol{x}$，返回两个变量——x 处的非线性不等式向量 $\boldsymbol{c}$ 和 x 处的非线性等式向量 $\boldsymbol{ceq}$。例如，若 nonlcon='mycon'，则 M 文件的形式如下：

```
function [c,ceq] = mycon(x)
c = ...     % 计算 x 处的非线性不等式
ceq = ...   % 计算 x 处的非线性等式
```

若约束函数的梯度可以计算，且 options.GradConstr 设为'on',即

```
options = optimset('GradConstr','on')
```

则函数 nonlcon 也必须在第 3 个和第 4 个输出变量中输出 $c(x)$的梯度 GC 和 $ceq(x)$的梯度 GCeq。注意，可以通过核对 nargout 参数来避免计算 GC 和 GCeq。

```
function [c,ceq,GC,GCeq] = mycon(x)
c = ...          % x 处的非线性不等式
ceq = ...        % x 处的非线性等式
if nargout > 2   % 被调用的 nonlcon 函数，有 4 个输出
   GC = ...      % 不等式的梯度
   GCeq = ...    % 等式的梯度
end
```

若 nonlcon 函数返回 m 元素的向量 $\boldsymbol{c}$ 和长度为 n 的 x，则 $c(x)$的梯度 GC 是一个 $n\times m$ 的矩阵，其中 GC(i,j)是 $c(j)$对 $x(i)$的偏导数。同样，若 $\boldsymbol{ceq}$ 是一个 p 元素的向量，则 $\boldsymbol{ceq}(\boldsymbol{x})$的梯度 Gceq 是一个 $n\times p$ 的矩阵，其中 Gceq(i,j)是 $ceq(j)$对 $x(i)$的偏导数。

options 变量为优化参数选项。可以用 optimset 函数设置或改变这些选项：

- DerivativeCheck——比较用户提供的导数（目标函数或约束函数的梯度）和有限差分导数。
- Diagnostics——打印将要最小化或求解的函数的诊断信息。
- DiffMaxChange——变量中有限差分梯度的最大变化。
- DiffMinChange——变量中有限差分梯度的最小变化。

- Display——显示水平。设置为'off' 时不显示输出；设置为 'iter' 时显示每一次迭代的输出；设置为'final'时只显示最终结果。
- GoalExactAchieve——使得目标个数刚好达到，不多也不少。
- GradConstr——用户定义的约束函数的梯度。
- GradObj——用户定义的目标函数的梯度。使用大型方法时必须使用梯度，对于中型方法则是可选项。
- MaxFunEvals——函数评价的允许最大次数。
- MaxIter——函数迭代的允许最大次数。
- MeritFunction——如果设为'multiobj'，则使用目标达到或最大最小化目标函数的方法。若设置为'singleobj'，则使用 fmincon 函数计算目标函数。
- TolCon——约束矛盾的终止容限。
- TolFun——函数值处的终止容限。
- TolX——x 处的终止容限。

weight 变量为权重向量，可以控制低于或超过 fgoalattain 函数指定目标的相对程度。当 goal 的值都是非零值时，为了保证激活对象超过或低于的比例相当，将权重函数设置为 abs(goal) (激活对象为阻止解处目标改善的对象集合)。当目标值中的任意一个为零时，设置 weight=abs(goal)将导致目标约束看起来更象硬约束，而不象目标约束。当加权函数 weight 为正时，fgoalattain 函数试图使对象小于目标值。为了使目标函数大于目标值，将权重 weight 设置为负。为了使目标函数尽可能地接近目标值，使用 GoalsExactAchieve 参数，将 fun 函数返回的第 1 个元素作为目标。

attainfactor 变量是超过或低于目标的个数。若 attainfactor 为负，则目标已经溢出；若 attainfactor 为正，则目标个数还未达到。

其他参数意义可以参见表 15-7 和表 15-8。

注意：当特征值为复数时，本问题不连续，这也说明了为什么收敛速度很慢。尽管原始方法假设函数是连续的，该法仍然可以向解的方向前进，因为在解的位置上，没有发生不连续的现象。当对象和目标为复数时，fgoalattain 函数将试图得到最小二乘意义上的目标。

多目标优化同时涉及到一系列对象。fgoalattain 函数求解该问题的基本算法是目标达到法。该法为目标函数建立起目标值。多目标优化的具体算法在前面进行了详细的介绍。具体实现过程中，使用了松弛变量 γ 作为模糊变量同时最小化目标向量 $F(x)$，goal 参数是一系列目标达到值。在进行优化之前，通常不知道对象是否会达到目标。使用权向量 weight 可以控制究竟是没有达到还是溢出。

fgoalattain 函数使用序列二次规划法(SQP)，前面已经进行了比较多的介绍。算法中对于一维搜索和 Hess 矩阵进行了修改。当有一个目标函数不再发生改善时，一维搜索终止。修改的 Hess 矩阵借助于本问题的结构，也被采用。

attainfactor 参数包含解处的 γ 值。γ 取负值时表示目标溢出。

目标函数必须是连续的。fgoalattain 函数将只给出局部最优解。

19.3 应用实例

【例 19-1】 某化工厂拟生产两种新产品 A 和 B, 其生产设备费用分别为：A，2 万元/吨；

B，5 万元/吨。这两种产品均将造成环境污染，设由公害所造成的损失可折算为：A，4 万元/吨；B，1 万元/吨。由于条件限制，工厂生产产品 A 和 B 的最大生产能力各为每月 5 吨和 6 吨，而市场需要这两种产品的总量每月不少于 7 吨。试问工厂如何安排生产计划，在满足市场需要的前提下，使设备投资和公害损失均达最小。该工厂决策认为，这两个目标中环境污染应优先考虑，设备投资的目标值为 20 万元，公害损失的目标为 12 万元。

设工厂每月生产产品 A 为 x_1 吨，B 为 x_2 吨，设备投资费为 $f_1(x)$，公害损失费为 $f_2(x)$，则这个问题可表达为多目标优化问题：

$$\begin{cases} \min f_1(\boldsymbol{x}) = 2x_1 + 5x_2 \\ \min f_2(\boldsymbol{x}) = 4x_1 + x_2 \\ x_1 \leqslant 5 \\ x_2 \leqslant 6 \\ x_1 + x_2 \geqslant 7 \\ x_1, x_2 \geqslant 0 \end{cases}$$

首先需要编写目标函数的 M 文件 opt19_1o.m，返回目标计算值

```
function f=myfun(x)
f(1)=2*x(1)+5*x(2);
f(2)=4*x(1)+x(2);
```

给定目标，权重按目标比例确定，给出初值

```
goal=[20 12];
weight=[20 12];
x0=[2 5];
```

给出约束条件的系数

```
A=[1 0;0 1;-1 -1];
b=[5 6 -7];
lb=zeros(2,1);
[x,fval,attainfactor,exitflag] = …
fgoalattain(@opt19_1o,x0,goal,weight,A,b,[],[],lb,[])
```

计算结果为

```
x =
    2.9167    4.0833
fval =
   26.2500   15.7500
attainfactor =
    0.3125
exitflag =
     1
```

故工厂每月生产产品 A 为 2.9167 吨，B 为 4.0833 吨。设备投资费和公害损失费的目标值分别为 26.2500 万元和 15.7500 万元。达到因子为 0.3125，计算收敛。

【例 19-2】 某厂生产两种产品 A 和 B，已知生产 A 产品 100kg 需 8 个工时，生产 B 产品 100kg 需 10 个工时。假定每日可用的工时数为 40，且希望不雇临时工，也不加班生产。这两种产品每 100kg 均可获利 100 元。此外，有个顾客要求每日供应 B 种产品 600kg。问应如何安排生产计划？

设生产 A，B 两种产品的数量分别为 x_1 和 x_2(均以 100kg 计)，为了使生产计划比较合理，要求用人尽量少，获利尽可能多，另外 B 种产品的产量尽量多。由题意建立下面的数学模型：

$$\begin{cases} \min z_1 = 8x_1 + 10x_2 \\ \max z_2 = 100x_1 + 100x_2 \\ \max z_3 = x_2 \\ 8x_1 + 10x_2 \leqslant 40 \\ x_2 \geqslant 6 \\ x_1, x_2 \geqslant 0 \end{cases}$$

首先需要编写目标函数的 M 文件 opt19_2o.m，返回目标计算值

```
function f=myfun(x)
f(1)=8*x(1)+10*x(2);
f(2)=-100*x(1) -100*x(2);
f(3)=-x(2);
```

给定目标，权重按目标比例确定，给出初值

```
goal=[40 -800 -6];
weight=[40 -800 -6];
x0=[2 2];
```

给出约束条件的系数

```
A=[8 10;0 -1];
b=[40 -6];
lb=zeros(2,1);
options=optimset('MaxFunEvals',5000);  % 设置函数评价的最大次数为 5000 次
[x,fval,attainfactor,exitflag] = …
fgoalattain(@opt19_2o,x0,goal,weight,A,b,[],[],lb,[],[], options);
```

计算结果为

```
x =
    2.0429    1.9458
fval =
   35.8007 -398.8648   -1.9458
attainfactor =
   -0.0646
exitflag =
     0
```

经过 5000 次迭代以后，生产 A，B 两种产品的数量各为 204.29kg 和 194.58kg。

【例 19-3】 某工厂因生产需要欲采购一种原材料，市场上的这种原料有两个等级，甲级单价 2 元/kg，乙级单价 1 元/kg。要求所花总费用不超过 200 元，购得原料总量不少于 100kg，其中甲级原料不少于 50kg，问如何确定最好的采购方案。

设 x_1、x_2 分别为采购甲级和乙级原料的数量（kg），要求采购总费用尽量少，采购总重量尽量多，采购甲级原料尽量多。由题意可得：

$$
\begin{cases}
\min z_1 = 2x_1 + x_2 \\
\max z_2 = x_1 + x_2 \\
\max z_3 = x_1 \\
2x_1 + x_2 \leqslant 200 \\
x_1 + x_2 \geqslant 100 \\
x_1 \geqslant 50 \\
x_1, x_2 \geqslant 0
\end{cases}
$$

首先需要编写目标函数的 M 文件 opt19_3o.m，返回目标计算值

```
function f=myfun(x)
f(1)=2*x(1)+ x(2);
f(2)=-x(1)- x(2);
f(3)=-x(1);
```

给定目标，权重按目标比例确定，给出初值

```
goal=[200 –100 -50];
weight=[2040 –100 -50];
x0=[55 55];
```

给出约束条件的系数

```
A=[2 1;-1 –1;-1 0];
b=[200 -100 -50];
lb=zeros(2,1);
[x,fval,attainfactor,exitflag] = …
fgoalattain(@opt19_3o,x0,goal,weight,A,b,[],[],lb,[]);
```

输出计算结果

```
x =
    50    50
fval =
   150  -100   -50
attainfactor =
  1.3235e-023
exitflag =
     1
```

所以，最好的采购方案是采购甲级原料和乙级原料各 50kg。此时采购总费用为 150 元，总重量为 100kg，甲级原料总重量为 50kg。

第 20 章　最大最小化

20.1　算法

通常我们遇到的都是目标函数的最大化和最小化问题，但是在某些情况下，则要求使最大值最小化才有意义。例如城市规划中需要确定急救中心、消防中心的位置，可取的目标函数应该是到所有地点最大距离的最小值，而不是到所有目的地的距离和为最小。这是两种完全不同的准则，在控制理论、逼近论、决策论中也使用最大最小化原则。

最大最小化问题的数学模型为

$$
\begin{aligned}
&\min_{x} \max_{\{F_1\}} \{F_1(\boldsymbol{x})\} \\
&\boldsymbol{c}(\boldsymbol{x}) \leqslant 0 \\
&\boldsymbol{ceq}(\boldsymbol{x}) = 0 \\
&\boldsymbol{A} \cdot \boldsymbol{x} \leqslant \boldsymbol{b} \\
&\boldsymbol{Aeq} \cdot \boldsymbol{x} = \boldsymbol{beq} \\
&\boldsymbol{lb} \leqslant \boldsymbol{x} \leqslant \boldsymbol{ub}
\end{aligned}
$$

式中 $\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{beq}, \boldsymbol{lb}$ 和 $\boldsymbol{ub}$ 为向量，$\boldsymbol{A}$ 和 $\boldsymbol{Aeq}$ 为矩阵，$\boldsymbol{c}(\boldsymbol{x})$, $\boldsymbol{ceq}(\boldsymbol{x})$和 $F(\boldsymbol{x})$为函数，返回向量。$F(\boldsymbol{x})$, $\boldsymbol{c}(\boldsymbol{x})$和 $\boldsymbol{ceq}(\boldsymbol{x})$可以是非线性函数。

MATLAB 优化工具箱中采用序列二次规划法求解最大最小化问题。

20.2　有关函数介绍

fminimax 使多目标函数中的最坏情况达到最小化。给定初值估计，该值必须服从一定的约束条件。其调用格式为：

● x = fminimax(fun,x0)初值为 x0，找到 fun 函数的最大最小化解 *x*。

● x = fminimax(fun,x0,A,b)给定线性不等式 A*x <= b，求解最大最小化问题。

● x = fminimax(fun,x,A,b,Aeq,beq) 给定线性等式，Aeq*x = beq，求解最大最小化问题。如果没有不等式存在，则设置 A=[ ]、b=[ ]。

● x = fminimax(fun,x,A,b,Aeq,beq,lb,ub) 为设计变量定义一系列下限 lb 和上限 ub，使得总有 lb <= x <= ub。

● x = fminimax(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon) 在 nonlcon 参数中给定非线性不等式约束 *c*(*x*)或等式约束 *ceq*(*x*)，fminimax 函数要求 c(x) <= 0 且 ceq(x) = 0。若没有边界存在，则设置 lb=[ ]和（或）ub=[ ]。

● x = fminimax(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options) 用 options 给定的参数进行优化。

● x = fminimax(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options,P1,P2,...)将问题参数 P1, P2 等直接传递给函数 fun 和 nonlcon。如果不需要变量 A, b, Aeq, beq, lb, ub, nonlcon 和 options 则将它们设置为空矩阵。

- [x,fval] = fminimax(...) 返回解 x 处的目标函数值。
- [x,fval,maxfval] = fminimax(...) 返回解 x 处的最大函数值。
- [x,fval,maxfval,exitflag] = fminimax(...) 返回 exitflag 参数，描述函数计算的退出条件。
- [x,fval,maxfval,exitflag,output] = fminimax(...) 返回描述优化信息的结构输出 output 参数。
- [x,fval,maxfval,exitflag,output,lambda] = fminimax(...) 返回包含解 x 处拉格朗日乘子的 lambda 参数。

调用格式中的 maxfval 变量为解 x 处函数值的最大值，即，maxfval = max{fun(x)}。

fminimax 函数使用序列二次规划法(SQP)进行计算。对一维搜索法和 Hess 矩阵的计算进行了修改。在一维搜索中，将精确目标函数和另外目标函数一起使用。当有一个目标函数不再发生改善时，一维搜索终止。修改的 Hess 矩阵借助于本问题的结构，也被采用。

使用 fminmax 函数时需要注意下面几个问题：

（1）在 options.MinAbsMax 中设置 F 最大绝对值最小化了的目标数。该目标应该放到 F 的第 1 个元素中去。例如，考虑上面的问题，需要找到 x 值，使下式的最大绝对值最小化：

$$\left[f_1(x), f_2(x), f_3(x), f_4(x), f_5(x)\right]$$

通过调用 fminimax 函数来求解

```
x0 = [0.1; 0.1];   % 设置初值
options = optimset('MinAbsMax',5);   % 使最大绝对值最小化
[x,fval] = fminimax(fun,x0,[],[],[],[],[],[],[],options);
```

经过 7 次迭代以后，得到问题的解

```
x =
    4.9256
    2.0796
fval =
    37.2356  -37.2356  -6.8357  -7.0052  -0.9948
```

（2）当提供了等式约束并且在二次子问题中发现并剔除了因变等式时，则在过程标题中打印'dependent'字样(当输出选项设置为 options.Display='iter')。因变等式只有在等式连续的情况下才被剔除。若系统不连续，则子问题不可行并且在过程标题中打印'infeasible'字样。

另外，要求目标函数必须连续，否则 fminimax 函数有可能给出局部最优解。

20.3 应用实例

【例 20-1】 定位问题。

设某城市有某种物品的 10 个需求点，第 i 个需求点 P_i 的坐标为（a_i, b_i），道路网与坐标轴平行，彼此正交。现打算建一个该物品的供应中心，且由于受到城市某些条件的限制，该供应中心只能设在 $\boldsymbol{x}$ 界于[5，8]，y 界于[5，8]的范围内。问该中心应建在何处为好？

P_{i} 点的坐标为：

a_{i}: 1	4	3	5	9	12	6	20	17	8
b_{i}: 2	10	8	18	1	4	5	10	8	9

设供应中心的位置为（x, y），要求它到最远需求点的距离尽可能小。由于此处应采用沿道路行走的距离，可知用户 P_i 到该中心的距离为 $|x-a_i|+|y-b_i|$，从而可得目标函数如下：

$$\min_{x,y}\{\max_{1\leqslant i\leqslant m}[|x-a_i|+|y-b_i|]\}$$

约束条件为

$$\begin{cases} x\geqslant 5 \\ x\leqslant 8 \\ y\geqslant 5 \\ y\leqslant 8 \end{cases}$$

首先，编写一个计算 x 处 10 个目标函数的 M 文件 opt20_1o.m。

```
function f = myfun(x)
%输入各个点的坐标值
a=[1  4  3  5  9  12  6  20  17  8];
b=[2  10 8  18 1  4   5  10  8   9];
f(1) = abs(x(1)-a(1))+abs(x(2)-b(1));
f(2) = abs(x(1)-a(2))+abs(x(2)-b(2));
f(3) = abs(x(1)-a(3))+abs(x(2)-b(3));
f(4) = abs(x(1)-a(4))+abs(x(2)-b(4));
f(5) = abs(x(1)-a(5))+abs(x(2)-b(5));
f(6) = abs(x(1)-a(6))+abs(x(2)-b(6));
f(7) = abs(x(1)-a(7))+abs(x(2)-b(7));
f(8) = abs(x(1)-a(8))+abs(x(2)-b(8));
f(9) = abs(x(1)-a(9))+abs(x(2)-b(9));
f(10) = abs(x(1)-a(10))+abs(x(2)-b(10));
```

然后，输入初值、约束条件并调用优化过程进行计算：

```
x0 = [6; 6];      % 提供解的初值
AA=[-1 0
   1  0
   0 -1
   0  1];
bb=[-5; 8; -5; 8];
[x,fval] = fminimax(@opt20_1o,x0,AA,bb)
```

计算结果为：

```
x =
   8
   8
fval =
   13   6   5   13   8   8   5   14   9   1
```

可见，在限制区域内的东北角设置供应中心可以使该点到各需求点的最大距离最小。最小的最大距离为 14 个距离单位。

第 21 章　半无限问题

21.1　基本数学原理

半无限约束问题的数学模型如下所示：

$$
\begin{aligned}
&\min_{x} f(\boldsymbol{x}) \\
&\boldsymbol{c}(\boldsymbol{x}) \leqslant 0 \\
&\boldsymbol{ceq}(\boldsymbol{x}) = 0 \\
&\boldsymbol{A} \cdot \boldsymbol{x} \leqslant \boldsymbol{b} \\
&\boldsymbol{Aeq} \cdot \boldsymbol{x} = \boldsymbol{beq} \\
&\boldsymbol{lb} \leqslant \boldsymbol{x} \leqslant \boldsymbol{ub} \\
&\boldsymbol{K}_1(\boldsymbol{x}, \boldsymbol{w}_1) \leqslant 0 \\
&\boldsymbol{K}_2(\boldsymbol{x}, \boldsymbol{w}_2) \leqslant 0 \\
&\cdots \\
&\boldsymbol{K}_n(\boldsymbol{x}, \boldsymbol{w}_n) \leqslant 0
\end{aligned}
$$

其中，$\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{beq}, \boldsymbol{lb}$ 和 $\boldsymbol{ub}$ 为向量，$\boldsymbol{A}$ 和 $\boldsymbol{Aeq}$ 为矩阵, $\boldsymbol{c}(\boldsymbol{x})$, $\boldsymbol{ceq}(x)$和 $\boldsymbol{K_i}(\boldsymbol{x},\boldsymbol{w_i})$为返回向量的函数。$f(\boldsymbol{x})$为返回标量的函数。$f(\boldsymbol{x})$, $\boldsymbol{c}(\boldsymbol{x})$和 $\boldsymbol{ceq}(\boldsymbol{x})$可以是非线性函数。向量（或矩阵）$\boldsymbol{K}_1(\boldsymbol{x},\boldsymbol{w}_1)\leqslant 0$即为半无限约束，它是 $\boldsymbol{x}$ 和其他变量 $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots \boldsymbol{w}_n$ 的连续函数。$\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots \boldsymbol{w}_n$ 变量的最大长度为 2。

MATLAB 优化工具箱采用二次、三次混合插值法结合序列二次规划法（SQP）进行问题求解。

21.2　有关函数介绍

用 fseminf 函数求半无限约束多变量非线性函数的最小值。其调用格式为：

- fseminf 函数求几个变量半无限约束标量函数的最小值，初值给定。目标是使 $f(x)$最小化。因为不可能计算所有的可能值，所以必须选择一个区域，在它上面计算样本数据集。
- x = fseminf(fun, x0, ntheta, seminfcon) 初值为 x0，求约束条件为 ntheta，半无限约束为 seminfcon 的 fun 函数的最小值。
- x = fseminf(fun, x0, ntheta, seminfcon, A, b) 该函数试图满足线性不等式 A*x <= b。
- x = fseminf(fun, x0, ntheta, seminfcon, A, b, Aeq, beq) 在上面的基础上添加线性等式 Aeq*x = beq。当没有不等式存在时，设置 A=[ ]、b=[ ]。
- x = fseminf(fun, x0, ntheta, seminfcon, A, b, Aeq, beq, lb, ub) 定义设计变量 x 的一系列下界 lb 和上界 ub, 使得总有 lb <= x <= ub。
- x = fseminf(fun, x0, ntheta, seminfcon, A, b, Aeq, beq, lb, ub, options) 用 options 结构指定的优化参数进行最小化。

- x = fseminf(fun, x0, ntheta, seminfcon, A, b, Aeq, beq, lb, ub, options, P1, P2,...) 将问题参数 P1、P2 等直接传递给函数 fun 和 seminfcon。若变量 A, b, Aeq, beq, lb, ub 和 options 不需要，则将它们设置为空矩阵。
- [x, fval] = fseminf(...) 返回解 x 处的目标函数值。
- [x, fval, exitflag] = fseminf(...) 返回描述退出条件的 exitflag 参数。
- [x, fval, 同 exitflag, output] = fseminf(...) 返回包含优化信息的输出参数 output。
- [x, fval, exitflag, output, lambda] = fseminf(...) 返回包含解 x 处拉格朗日乘子的 lambda 参数。

各调用格式中，ntheta 参数为半无限约束的个数。options 参数为优化参数选项，有以下一些属性：

- DerivativeCheck ——比较用户提供的导数（梯度）和有限差分导数。
- Diagnostics ——打印待最小化或待求解函数的诊断信息。
- DiffMaxChange ——变量有限差分的最大变化。
- DiffMinChange ——变量有限差分的最小变化。
- Display ——显示水平。设置为 'off' 时不显示输出；设置为 'iter' 时显示每一步迭代过程的输出；设置为 'final' 时显示最终的迭代结果。
- GradObj ——用户定义的目标函数的梯度信息。对于大型算法，必须提供梯度；对于中型算法，梯度则是可选项。
- MaxFunEvals ——函数评价的最大次数。
- MaxIter ——函数迭代的允许最大次数。
- TolCon ——约束矛盾的终止容限。
- TolFun ——函数值的终止容限。
- TolX ——解 x 处的终止容限。

seminfcon 参数计算非线性不等式向量 ***c***，非线性等式向量 ***ceq*** 和 **ntheta** 半无限约束（向量或矩阵）K1, K2,..., Kntheta。seminfcon 参数是一个包含函数名的字符串，该函数可以是 M 文件、内部文件或 MEX 文件。例如，若 seminfcon='myinfcon'，则 M 文件 myinfcon.m 有下面的形式：

```
function [c,ceq,K1,K2,...,Kntheta,S] = myinfcon(x,S)
% 初始化样本区间
if isnan(S(1,1)),
   S = ...       % S 有 ntheta 行、2 列
end
w1 = ...         % 计算样本集
w2 = ...         % 计算样本集
...
wntheta = ...    % 计算样本集
K1 = ...         % x 和 w 处的一阶半无限约束
K2 = ...         % x 和 w 处的二阶半无限约束
...
Kntheta = ...    % x 和 w 处的最后一阶半无限约束
c = ...          % 计算 x 处的非线性不等式
ceq = ...        % 计算 x 处的非线性等式
```

S 为建议的样本区间，它可能会被利用，也可能不被利用。如果没有此约束存在，将为 ***c*** 和 ***ceq*** 返回空矩阵。

向量或矩阵 $\boldsymbol{K_1}$, $\boldsymbol{K_2}$, ..., ***Kntheta*** 包含分别为样本集中的独立变量 w_1, w_2, ... wntheta 进行评价的半无限约束。两个列矩阵 ***S*** 包含 w_1, w_2, ... wntheta 值的建议的样本区间，它们用于评价 $\boldsymbol{K_1}$, $\boldsymbol{K_2}$, ... , ***Kntheta***。***S*** 矩阵的第 *i* 行包含评价 $\boldsymbol{K_i}$ 的样本区间。当 $\boldsymbol{K_i}$ 为一向量时，只用 $\boldsymbol{S}(i, j)$（第 2 列可以都为零）。当 $\boldsymbol{K_i}$ 为一矩阵时，$\boldsymbol{S}(i,\ 2)$用于对 $\boldsymbol{K_i}$ 中的行进行取样，$\boldsymbol{S}(i,1)$被用作 $\boldsymbol{K_i}$ 中列的取样区间。第一次迭代，***S*** 为空值，所以有些初始取样区间必须由 seminfcon 参数来决定。

其他参数的意义可以参见表 15-7 和表 15-8。

fseminf 函数使用二次和三次混合插值法估计半无限约束范围内的峰值。峰值用于组成一系列约束提供给 SQP 法，就像在 fmincon 函数中所做的一样。当约束的数目改变时，拉格朗日乘子被重新分配到新的约束集合中去。

建议计算取样区间时使用内插峰值和峰值之间的差值去估计插值点的数目。内插效果通过曲线外推和与曲线中其他点比较而被考虑。当峰值接近约束边界时，建议的取样区间变窄了。

使用 fseminf 函数时需要注意下面一些问题：

- seminfcon 参数中设置的建议取样区间 ***S*** 可能会被计算过程中的优化方法改变，因为其他不是建议值的值可能会更有效或更稳健。所以，计算 $\boldsymbol{K}_i(x_1 w_i)$ 时，只要不引起局部极小值的明显变化，其中的 w_i 是允许变化的。
- 目标函数和约束函数，以及半无限约束函数必须为 ***x*** 和 ***w*** 的连续函数。fseminf 函数可能只给出局部最优解。
- 当问题不可行时，fseminf 函数试图使最大的约束值最小化。

21.3　应用实例

【例 21-1】一维问题。

假设有下面的函数：

$$f(x) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 + (x_3 - 0.5)^2$$

式中

$$K_1(x, w_1) = \sin(w_1 x_1)\cos(w_1 x_2) - \frac{1}{1000}(w_1 - 50)^2 - \sin(w_1 x_3) - x_3 \leqslant 1$$

$$K_1(x, w_2) = \sin(w_2 x_2)\cos(w_2 x_2) - \frac{1}{1000}(w_2 - 50)^2 - \sin(w_2 x_3) - x_3 \leqslant 1$$

w_1 和 w_2 有下面的约束：

$$1 \leqslant w_1 \leqslant 100$$

$$1 \leqslant w_2 \leqslant 100$$

注意：这里半无限约束是一维向量。因为约束条件必须写为 $K_1(x, w_1) \leqslant 0$ 的形式，故需要对约束做如下计算。

首先，写一个 M 文件 opt21_1o.m，计算目标函数值。

```
function f = myfun(x, s)
% 目标函数
f = sum((x-0.2).^2);
```

然后，编写 M 文件 opt21_1c.m，计算非线性等式约束和非线性不等式约束，以及半无限

约束：

```
function [c,ceq,K1,K2,s] = mycon(x,s)
% 初始化样本区间
if isnan(s(1,1)),
   s = [0.2 0; 0.2 0];
end
% 样本集
w1 = 1:s(1,1):100;
w2 = 1:s(2,1):100;
% 半无限约束
K1 = sin(w1*x(1)).*cos(w1*x(2))-1/1000*(w1-50).^2 -...
     sin(w1*x(3))-x(3)-1;
K2 = sin(w2*x(2)).*cos(w2*x(1))-1/1000*(w2-50).^2 -...
     sin(w2*x(3))-x(3)-1;
% 无约束
c = [];  ceq=[];
% 绘制半无限约束图
plot(w1, K1,'-', w2, K2, :'),title('Semi-infinite constraints')
drawnow
```

然后调用优化过程：

```
x0 = [0.5; 0.2; 0.3];     % 给初值
[x,fval] = fseminf(@opt21_1o, x0, 2, @opt21_1c)
```

经过 8 次迭代以后，得到问题的解

```
x =
    0.6673
    0.3013
    0.4023
```

解 x 处的函数值和半无限约束的最大值为

```
fval =
    0.0770
[c,ceq,K1,K2] = opt28_1c(x,NaN);
max(K1)
ans =
    -0.0017
max(K2)
ans =
    -0.0845
```

生成半无限约束图，如图 21-1 所示。

该图演示了约束边界上两个函数如何达到峰值。

'mycon.m' 文件内部的绘图语句将使计算速度降低，剔除它可以加快速度。

【例 21-2】二维问题。

该问题的数学模型为

$$f(\boldsymbol{x})=(x_1-0.5)^2+(x_2-0.5)^2+(x_3-0.5)^2$$

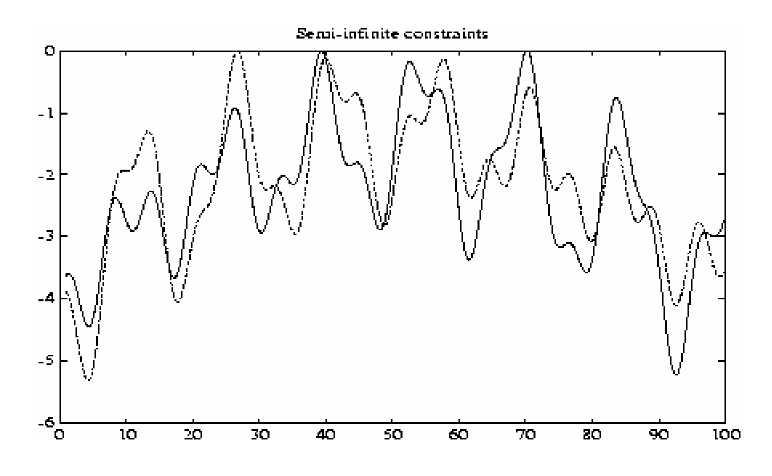


图 21-1　一维问题的半无限约束图

式中

$$K_1(\boldsymbol{x},\boldsymbol{w}) = \sin(w_1 x_1)\cos(10 w_2 x_2) - \frac{1}{1000}(w_1 - 50)^2 - \sin(10 w_1 x_3) - x_3 + \ldots$$

$$\sin(w_2 x_2)\cos(w_1 x_1) - \frac{1}{1000}(w_2 - 50)^2 - \sin(w_2 x_3) - x_3 \leqslant 1.5$$

w_1 和 w_2 有下面的约束：

$$1 \leqslant w_1 \leqslant 100$$

$$1 \leqslant w_2 \leqslant 100$$

初始点为 $\boldsymbol{x}$=[0.2　0.2　0.2]。

注意半无限约束是二维的，即为矩阵。

首先，将上例中的 M 文件 opt21_1o.m 再用一遍。

第 2 步，为约束条件编写 M 文件 opt21_2c.m：

```
function [c, ceq, K1,s] = mycon(x, s)
% 初始化取样区间
if isnan(s(1, 1)),
   s = [2 2];
end
% 样本集
w1x = 1:s(1,1):100;
w1y = 1: s(1, 2):100;
[wx,wy]=meshgrid(w1x,w1y);
%
% 半无限约束
K1=sin(wx*x(1)).*cos(wy*x(2))-1/1000*(wx-50).^2 -...
    sin(wx*x(3))-x(3)+sin(wy*x(2)).*cos(wx*x(1))-...
    1/1000*(wy-50).^2-sin(wy*x(3))-x(3)-1.5;
% 无有限非线性约束
c=[]; ceq=[];
% 网格图
mesh(K1)
title('Semi-infinite constraint')
```

```
drawnow
```

下一步调用优化过程(M 文件为 opt21_2.m)

```
x0 = [0.25, 0.25, 0.25];    % 初值
[x,fval] = fseminf(@opt21_1o,x0,1,@opt21_2c)
```

经过 7 次迭代以后，得到问题的解：

```
x =
    0.2081    0.2066    0.1965
[c, ceq, K1] = opt28_2c(x, NaN);
```

解 x 处的函数值和半无限约束的最大值为

```
fval =
    1.2066e-04
max(max(K1))
ans =
   -0.0262
```

生成网格图如图 21-2 所示。

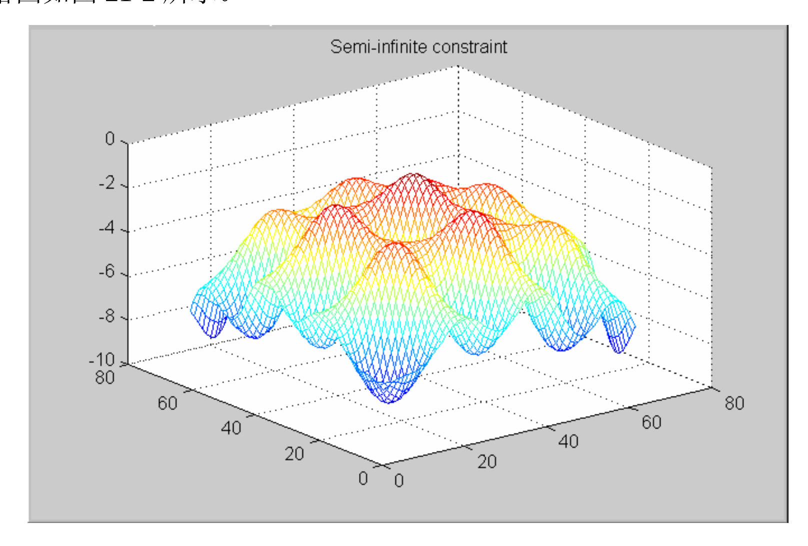


图 21-2 二维问题的半无限约束图

第 22 章　最小二乘问题

22.1　算法

最小二乘问题可用下式表达：

$$\min_{x\in R^n} f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{F}(\boldsymbol{x})\|_2^2 = \frac{1}{2}\sum_i \boldsymbol{F}_i(\boldsymbol{x})^2$$

此类问题在实际应用中很常见，特别是进行数据拟合、非线性参数估计时。最小二乘问题的常用解法有 Gauss-Newton 法和 Levenberg-Marquardt 法。

1．Gauss-Newton 法

该法通过在每一次迭代中求解下列线性最小二乘问题来获得搜索方向 d_k。

$$\min_{x\in R^n}\|\boldsymbol{J}(\boldsymbol{x}_k)\boldsymbol{d}_k - \boldsymbol{F}(\boldsymbol{x}_k)\|_2^2$$

搜索方向 d_k 可以用于一维搜索，以保证每一次迭代都使函数 $f(x)$减小。

图 22-1 演示了采用 Gauss-Newton 法求解 Rosenbrock 函数最小化问题的路径。该法只用了 48 次迭代即完成计算。

Gauss-Newton 法有时会出现矩阵求逆的困难或出现假收敛的情况。用 Levenberg-Marquart 法可以解决此问题。

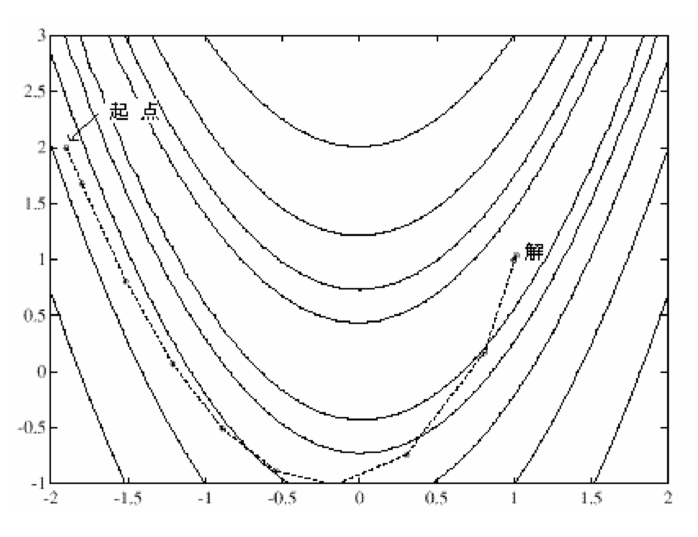


图 22-1　Gauss-Newton 法的搜索路径

2．Levenberg-Marquart 法（又称阻尼最小二乘法）

该法用下式求搜索方向：

$$(\boldsymbol{J}(\boldsymbol{x}_k)^T \boldsymbol{J}(\boldsymbol{x}) + \lambda_k \boldsymbol{I})\boldsymbol{d}_k = -\boldsymbol{J}(\boldsymbol{x}_k)\boldsymbol{F}(\boldsymbol{x}_k)$$

式中λ_k为阻尼因子，它可以控制 $\boldsymbol{d}_k$ 的大小和方向。当λ_k=0 时，即为 Gauss-Newton 法。当$\lambda_k \to \infty$时，趋于零向量，即为最速下降法。因此，只要给一个足够大的λ_k，$F(x_k+d_k)<F(x_k)$就始终为真。因此，即使是遇到影响 Gauss-Newton 法有效性的病态二次项，也可以通过阻尼因子λ_k来进行控制。

因此，Levenberg-Marquart 法给出的是介于 Gauss-Newton 法和最速下降法之间的搜索方向。图 22-2 是该法的演示，它用了 90 次迭代运算，介于 Gauss-Newton 法的 48 次和最速下降法的 1000 次之间。

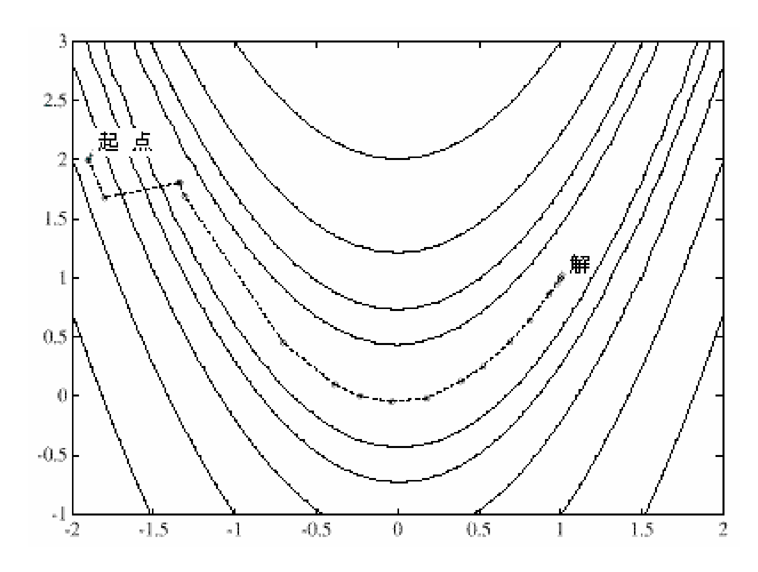


图 22-2　Levenberg-Marquart 法的搜索路径

22.2　线性最小二乘问题

利用左除（“\”）算子可以求解线性最小二乘问题。

如果 $\boldsymbol{A}$ 为平方矩阵，则除了算法不同，$\boldsymbol{A}\backslash\boldsymbol{b}$ 与 inv($\boldsymbol{A}$)*$\boldsymbol{b}$ 大致相当。如果 $\boldsymbol{A}$ 是一个 $n\times n$ 的矩阵，$\boldsymbol{b}$ 是一个具有 n 个元素的列向量或具有多个此类列向量的矩阵，则 $\boldsymbol{X}=\boldsymbol{A}\backslash\boldsymbol{b}$ 为用高斯消元法得到的方程 $\boldsymbol{AX}=\boldsymbol{b}$ 的解。如果 $\boldsymbol{A}$ 奇异，则会给出警告消息。

如果 $\boldsymbol{A}$ 为 $m\times n$ 的矩阵（$m\neq n$），且 $\boldsymbol{b}$ 为具有 m 个元素的列向量或具有多个此类列向量的矩阵，则 $\boldsymbol{X}=\boldsymbol{A}\backslash\boldsymbol{b}$ 为等式系统 $\boldsymbol{AX}=\boldsymbol{b}$ 的最小二乘意义上的解。

22.3　非负线性最小二乘解问题

22.3.1　基本数学原理

非负线性最小二乘解问题的数学模型为

$$\min_{x}\frac{1}{2}\|\boldsymbol{Cx}-\boldsymbol{d}\|_2^2$$
$$\boldsymbol{x}\geqslant 0$$

式中，矩阵 $\boldsymbol{C}$ 和向量 $\boldsymbol{d}$ 为目标函数的系数。独立变量向量 $\boldsymbol{x}$ 要求非负。

22.3.2　有关函数介绍

用 lsqnonneg 函数求线性问题的非负最小二乘解。其调用格式为：

● x = lsqnonneg(C,d) 返回向量 x，使范数(C*x-d)最小化，约束条件为 x >= 0。C 和 d 必须为实数。

● x = lsqnonneg(C,d,x0) 若所有的 x0>=0，则使用 x0 为初值；否则使用默认值。默认的初值为原点（当 x0==[ ]或只提供两个输入变量时也使用默认值）。

● x = lsqnonneg(C,d,x0,options) 用 options 结构指定的优化参数进行最小化。

● [x,resnorm] = lsqnonneg(...) 返回残差的平方范数值：norm(C*x-d)^2。

● [x,resnorm,residual] = lsqnonneg(...) 返回残差 C*x-d。

● [x,resnorm,residual,exitflag] = lsqnonneg(...) 返回 exitflag 参数，描述退出条件。

● [x,resnorm,residual,exitflag,output] = lsqnonneg(...) 返回包含优化信息的输出结构 output 参数。

● [x,resnorm,residual,exitflag,output,lambda] = lsqnonneg(...) 返回拉格朗日乘子 lambda。

各参数的意义可以参见表 15-7 和表 15-8。

lsqnonneg 函数使用的算法是从一系列可能的基本向量出发，计算相关的对偶向量 lambda。然后选择与 lambda 中最大值对应的基本向量并将它剔除，然后将下一个值交换进来。重复此过程，直到 lambda <= 0。

必须注意的是，非负最小二乘问题是有约束线性最小二乘问题的一个子集，所以当 ***C*** 的行数比列数多时，除了 lambda = -lambda_lsqlin.ineqlin 外，

```
[x,resnorm,residual,exitflag,output,lambda] = lsqnonneg(C,d)
```

等价于

```
[m,n] = size(C);
[x,resnorm,residual,exitflag,output,lambda_lsqlin] =
lsqlin(C,d,-eye(n,n),zeros(n,1));
```

对于大于 20 阶的问题，lsqlin 函数比 lsqnonneg 函数的计算速度快。否则，使用 lsqnonneg 函数更有效。

22.3.3　应用实例

【例 22-1】 利用下面的 4*2 问题比较无约束最小二乘解和 lsqnonneg 函数解。

```
C = [
      0.0372    0.2869
      0.6861    0.7071
      0.6233    0.6245
      0.6344    0.6170];
d = [
      0.8587
      0.1781
      0.0747
      0.8405];
[C\d, lsqnonneg(C,d)]
ans =
```

```
    -2.5627    0
     3.1108    0.6929
[norm(C*(C\d)-d), norm(C*lsqnonneg(C,d)-d)]
ans =
     0.6674 0.9118
```

二者的解不完全吻合，但非负最小二乘解没有为负的元素。

22.4　有约束线性最小二乘问题

22.4.1　基本数学原理

有约束线性最小二乘问题的数学模型为

$$\min_{x}\frac{1}{2}\|\boldsymbol{Cx}-\boldsymbol{d}\|_2^2$$
$$\boldsymbol{A}\cdot\boldsymbol{x}\leqslant\boldsymbol{b}$$
$$\boldsymbol{Aeq}\cdot\boldsymbol{x}=\boldsymbol{beq}$$
$$\boldsymbol{lb}\leqslant\boldsymbol{x}\leqslant\boldsymbol{ub}$$

式中，***C***, ***A*** 和 ***Aeq*** 为矩阵，***d***、***b***、***beq***、***lb***、***ub*** 和 ***x*** 为向量。

22.4.2　有关函数介绍

用 lsqlin 函数求解有约束线性最小二乘问题。调用格式为：

- x = lsqlin(C, d, A, b) 求解最小二乘意义上的线性系统 C*x=d，约束条件为 A*x<=b,其中 C 为 m*n 的矩阵。
- x = lsqlin(C, d, A,b, Aeq, beq) 加上等式约束 Aeq*x = beq 以后求解上面的问题。如果没有不等式存在，则令 A=[ ]、b=[ ]。
- x=lsqlin(C, d, A, b, Aeq, beq, lb, ub)为 x 定义一系列下界 *lb* 和上界 ub，使得总有 lb <=x<=ub。当没有等式约束存在时，令 Aeq=[ ]、beq=[ ]。x = lsqlin(C, d, A, b, Aeq, beq, lb, ub, x0) 设置初值 x0。
- x = lsqlin(C, d, A, b, Aeq, beq, lb, ub, x0, options) 用 options 结构指定的优化参数进行最小化。
- [x, resnorm] = lsqlin(...) 返回残差的平方范数值：norm(C*x-d)^2。
- [x, resnorm, residual] = lsqlin(...) 返回残差 C*x-d。
- [x, resnorm, residual, exitflag] = lsqlin(...) 返回 exitflag 参数，它包含函数的退出条件。
- [x, resnorm, residual, exitflag, output] = lsqlin(...) 返回与优化信息有关的结构输出参数 output。
- [x, resnorm, residual, exitflag, output, lambda] = lsqlin(...) 返回解 x 处包含拉格朗日乘子的 lambda。

各参数的意义可参见表 15-7 和表 15-8。

对于不同规模的问题，lsqlin 函数分别使用不同的算法：

- 大型优化算法——当优化问题只有上界或只有下界，即没有指定线性不等式或等式，而且矩阵 ***C*** 的行数至少与列数一样多时，默认的计算方法是大型优化算法。该法是一

种基于内部映射牛顿法的子空间置信域法。每一次迭代都与用 PCG 法求解大型线性系统得到的近似解有关。

- 中型优化算法——lsqlin 函数将 options.LargeScale 设置为 'off'，或者当给出了线性不等式或等式以后，使用活动集方法的 quadprog 函数。它首先通过求解线性规划问题来找到一个初始可行解。

另外，在使用 lsqlin 函数时还要注意下面这些问题：

- 对于无约束问题，应该用 \ 进行计算：***x***= ***A******b***。
- 一般，lsqlin 函数会找到一个局部解。
- 如果用 ***Aeq*** 和 ***Beq*** 指定等式约束，而不是用 ***lb*** 和 ***ub***，将会得到更好的数值解。
- 大型优化问题，若 ***x*0** 不是严格可行的，lsqlin 函数将选择一个新的严格可行的初始解。
- 若 x 的某些元素没有上界或下界，则 lsqlin 函数更希望 ***ub*** 或 ***lb*** 的对应元素被设置为 Inf 或-Inf，而不希望赋一个很大的数给 ***ub*** 或赋一个很小的负数给 ***lb***。

大型优化问题的优化代码中不允许出现约束上限与约束下限相等的情况。若有 ***lb***(2)==***ub***(2)，则给出下列出错消息：

Equal upper and lower bounds not permitted in this large-scale method.Use equality constraints and the medium-scale method instead.

此时，必须用中型优化算法求解等式约束问题。

对于中型优化问题，若矩阵 ***C***，***A*** 或 ***Aeq*** 为稀疏矩阵，而且用大型算法不能解决该问题，则 lsqlin 函数给出警告，说明矩阵将被转换为满秩矩阵：

Warning: This problem formulation not yet available for sparse matrices.Converting to full to solve.

当解不可行时，lsqlin 函数给出警告消息：

Warning: The constraints are overly stringent;there is no feasible solution.

本例中，lsqlin 生成使约束矛盾的最坏程度最小的结果。

当等式约束不连续时，lsqlin 函数给出警告：

Warning: The equality constraints are overly stringent;there is no feasible solution.

22.4.3　应用实例

【例 22-2】　求解下列问题的最小二乘解。

$$\boldsymbol{C}\cdot\boldsymbol{x}=\boldsymbol{d}$$
$$\boldsymbol{A}\cdot\boldsymbol{x}\leqslant\boldsymbol{b}$$
$$\boldsymbol{lb}\leqslant\boldsymbol{x}\leqslant\boldsymbol{ub}$$

首先输入系数矩阵和下界、上界：

```
C=[
    0.9501   0.7620   0.6153   0.4057
    0.2311   0.4564   0.7919   0.9354
    0.6068   0.0185   0.9218   0.9169
    0.4859   0.8214   0.7382   0.4102
    0.8912   0.4447   0.1762   0.8936];
d=[
    0.0578
```

```
    0.3528
    0.8131
    0.0098
    0.1388];
A=[
    0.2027    0.2721    0.7467    0.4659
    0.1987    0.1988    0.4450    0.4186
    0.6037    0.0152    0.9318    0.8462];
b=[
    0.5251
    0.2026
    0.6721];
lb = -0.1*ones(4,1);
ub = 2*ones(4,1);
```

然后调用有约束线性最小二乘过程：

```
[x,resnorm,residual,exitflag,output,lambda] = ...
lsqlin(C, d, A, b, [ ], [ ], lb, ub);
```

输入 x,lambda.ineqlin,lambda.lower,lambda.upper 得到

```
x =
   -0.1000
   -0.1000
    0.2152
    0.3502
lambda.ineqlin =
         0
    0.2392
         0
lambda.lower =
    0.0409
    0.2784
         0
         0
lambda.upper =
         0
         0
         0
         0
```

lambda 参数中的非零元素指示了解处的活动约束。本例中，lambda.ineqlin 中的第 2 个不等式和 lambda.lower 中的第 2 个下界都是活动约束（解在约束边界上）。

22.5　非线性最小二乘问题

22.5.1　基本数学原理

非线性最小二乘问题的数学模型为

$$\min_{x} f(\boldsymbol{x}) = f_1(\boldsymbol{x})^2 + f_2(\boldsymbol{x})^2 + \ldots + f_m(\boldsymbol{x})^2 + L$$

式中 L 为常数。

该问题主要通过 Gauss-Newton 法和 Levenberg-Marquardt 法进行求解。

22.5.2　有关函数介绍

用 lsqnonlin 函数求解非线性最小二乘问题，其调用格式为：

- lsqnonlin 函数求解非线性最小二乘问题，包括非线性数据拟合问题。
- 不仅仅要计算目标函数 $f(x)$ (平方和), lsqnonlin 函数还需要用户指定函数来计算向量值函数

$$\boldsymbol{F}(\boldsymbol{x})=\begin{bmatrix} f_1(\boldsymbol{x}) \\ f_2(\boldsymbol{x}) \\ f_3(\boldsymbol{x}) \end{bmatrix}$$

- 然后，将优化问题重新写成向量的形式：

$$\min_x \frac{1}{2}\|\boldsymbol{F}(\boldsymbol{x})\|_2^2 - \frac{1}{2}\sum_i \boldsymbol{f_i}(\boldsymbol{x})^2$$

式中，$\boldsymbol{x}$ 为一向量，$\boldsymbol{F}(\boldsymbol{x})$为一返回向量值的函数。

- x = lsqnonlin(fun, x0) 初值为 x0，求 fun 函数的最小平方和。fun 函数将返回一个数值向量但不是值的平方和。
- x = lsqnonlin(fun, x0, lb, ub) 定义一系列的下界 lb 和上界 ub，使得总有 lb <= x <= ub。
- x = lsqnonlin(fun, x0, lb, ub, options) 用 options 结构指定的优化参数进行最小化。
- x = lsqnonlin(fun, x0, lb, ub, options, P1, P2,...) 将问题参数 P1、P2 等直接传递给 fun 函数，将空矩阵传递给 options 参数作为其默认值。
- [x, resnorm] = lsqnonlin(...) 返回 x 处残差的平方范数值：　sum(fun(x).^2)。
- [x, resnorm, residual] = lsqnonlin(...) 返回解 x 处的残差值 fun(x)。
- [x, resnorm, residual, exitflag] = lsqnonlin(...) 返回 exitflag 参数，描述函数的退出条件。
- [x, resnorm, residual, exitflag, output] = lsqnonlin(...) 返回包含优化信息的输出结构参数 output。
- [x, resnorm, residual, exitflag, output, lambda] = lsqnonlin(...) 返回包含 x 处拉格朗日乘子的结构参数 lambda。
- [x, resnorm, residual, exitflag, output, lambda, jacobian] = lsqnonlin(...) 返回解 x 处的 fun 函数的雅可比矩阵。

各参数的意义可参见表 15-7 和表 15-8。

对于不同规模的问题，lsqnonlin 函数使用不同的算法：

- 大型优化问题——默认时，lsqnonlin 函数将选择大型优化算法。该算法是基于内部映射牛顿法的子空间置信域法。算法的每一次迭代步骤都涉及到用 PCG 法求解大型线性系统得到的近似解。
- 中型优化算法——lsqnonlin 函数通过将 options.LargeScale 设置为'off'来使用 Levenberg-Marquardt 法进行一维搜索。可通过设置 options.LevenbergMarquardt 参数来选择 Gauss-Newton 法进行一维搜索。设置的具体方法是将 options.LevenbergMarquardt 设置为'off' (且 options.LargeScale 设置为'off')。当残差较小时，使用 Gauss-Newton 法效果更好，速度更快。

默认的一维搜索法，如将 options.LineSearchType 设置为'quadcubic',将使用二次、三次混合插值法。将 options.LineSearchType 设置为'cubicpoly'，将采用三次多项式插值法进行一维搜索。该法一般需要较少的函数评价次数，但需要更多的梯度评价次数。这样，如果提供了梯度信息或梯度容易求得，则三次多项式插值是一个更佳的选择。

使用本函数时需要注意下面的问题：

- 大型优化问题的代码中不允许出现约束上界和约束下界相等的情况，若 lb(2)==ub(2)，将给出下面的出错消息：

  Equal upper and lower bounds not permitted.

- lsqnonlin 函数没有等式约束，有等式约束时需要用另外的方法。此时，fmincon 函数、fminimax 函数或 fgoalattain 函数都是可以选择的方法。
- 目标函数必须是连续的。lsqnonlin 函数可能只会给出局部最优解。
- lsqnonlin 函数只用于实数变量。当 x 为复数变量时，必须将它分解为实数部分和虚数部分。
- 大型优化问题——lsqnonlin 函数使用大型方法时不求解待定系统：它需要方程数（如，F 的元素个数）至少与变量的个数相等。对于待定系统，将使用中型算法进行求解。如果存在边界条件，将给出警告信息并且将忽略边界条件对问题进行求解。

若 x 的元素没有上界或下界，则 lsqnonlin 函数更希望 *ub* 或 *lb* 的对应元素设置为 Inf 或-Inf，而不希望给 *ub* 一个很大的正数而给 *lb* 一个很小的负数。

目前，若在 fun 函数中没有提供解析的雅可比矩阵，选项参数 DerivativeCheck 将不能与大型算法一起使用以比较解析的雅可比矩阵和有限差分得到的雅可比矩阵。取而代之，应该通过将 options 参数 MaxIter 设置为 0 次迭代来使用中型方法核对导数。然后使用大型方法求解问题。

- 中型优化问题——中型算法中要求没有边界约束。

由于大型算法不能求解待定系统，中型算法不能求解有边界约束的问题，所以具备这两个特征的优化问题都不能用 lsqnonlin 函数求解。

22.5.3　应用实例

【例 22-3】　求解 $\boldsymbol{x}$，使下式最小化：

$$\sum_{k=1}^{10}(2+2k-\mathrm{e}^{kx_1}-\mathrm{e}^{kx_2})^2$$

初值为 $\boldsymbol{x}=[0.3, 0.4]$。

因为 lsqnonlin 函数假设用户提供的平方和不是显式表达的，所以传递给 lsqnonlin 函数的函数应该计算向量值函数。

$$\boldsymbol{F}_k(x)=2+2k-\mathrm{e}^{kx_1}-\mathrm{e}^{kx_2}$$

k=1,2,…,10 (即，F 应该有 k 个元素)。

首先编写一个 M 文件 opt22_3o.m，计算 k 元素向量 $\boldsymbol{F}$：

```
function F = myfun(x)
k = 1:10;
F = 2 + 2*k-exp(k*x(1))-exp(k*x(2));
```

然后调用优化过程：

```
x0 = [0.3 0.4]          % 初值
[x,resnorm] = lsqnonlin(@opt22_3o,x0)    % 调用优化函数
```

经过 24 次函数评价以后，给出问题的解：

```
x =
    0.2578   0.2578
resnorm    %残差或平方和
resnorm =
    124.3622
```

【例 22-4】 求下面函数的极小点，初始点取为 **x0**=[1,1]。

$$f(x_1, x_2) = x_1^2 + 4x_2^2$$

首先编写目标函数的 M 文件 opt22_4o.m，

```
function F = myfun(x)
F = x(1)^2+4*x(2)^2;
```

然后调用优化过程：

```
x0 = [1  1];          % 初值
[x,resnorm]=lsqnonlin(@opt22_4o,x0)    % 调用优化函数
```

经过有限次数的迭代以后问题的解：

```
x =
   -0.0023   -0.0005
resnorm =
   3.6334e-011
```

可见该解非常接近于问题的正解[0, 0]。

22.6 非线性曲线拟合问题

22.6.1 基本数学原理

非线性曲线拟合问题的数学模型为

$$\min_x \frac{1}{2}\|\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{xdata}) - \boldsymbol{ydata}\|_2^2 - \frac{1}{2}\sum_i (\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{xdata}_i) - \boldsymbol{ydata}_i)^2$$

其中 ***xdata*** 和 ***ydata*** 为向量，***F***(***x, xdata***)为向量值函数。

MATLAB 用 lsqcurvefit 函数进行非线性曲线的拟合。其算法与 lsqnonlin 函数的算法相同。设置该函数的目的是提供一个更方便于进行数据拟合的方法。

22.6.2 有关函数介绍

用 lsqcurvefit 函数求解最小二乘意义上的非线性曲线拟合（数据拟合）问题。即根据输入数据 ***xdata*** 和得到的输出数据 ***ydata***，找到与方程 ***F***(***x, xdata***)最佳的拟合系数。该函数的调用格式为：

● lsqcurvefit 求解非线性数据拟合问题。lsqcurvefit 需要一个用户定义函数来计算向量值函数 ***F***(***x, xdata***)。用户定义的函数的向量的大小必须与 ***ydata*** 的大小相同。

● x = lsqcurvefit(fun, x0, xdata, ydata) 初值为 x0，求非线性函数 fun(x, xdata)与数据 ydata 在最小二乘意义上的拟合系数 x。ydata 的大小必须与 fun 函数返回的 F 向量或矩阵

的大小相同。

- x = lsqcurvefit(fun, x0, xdata, ydata, lb, ub) 定义解 x 的一系列下界 lb 和上界 ub，使得总有 lb <= x <= ub。
- x = lsqcurvefit(fun, x0, xdata, ydata, lb, ub, options) 用 options 参数指定的优化参数进行最小化。
- x = lsqcurvefit(fun, x0, xdata, ydata, lb, ub, options, P1, P2,...) 将问题参数 P1, P2 等直接传递给 fun 函数。options 参数的默认设置为空矩阵。
- [x,resnorm] = lsqcurvefit(...) 返回 *x* 处残差的平方和范数值：

sum{(fun(x, xdata)–ydata).^2}

- [x, resnorm, residual] = lsqcurvefit(...) 返回解 x 处的残差值：fun(x, xdata)-ydata。
- [x, resnorm, residual, exitflag] = lsqcurvefit(...) 返回描述退出条件的 exitflag 参数。
- [x, esnorm, residual, exitflag, output] = lsqcurvefit(...) 返回包含优化参数的输出结构参数 output。
- [x, resnorm, residual, exitflag, output, lambda] = lsqcurvefit(...) 返回包含解 x 处拉格朗日乘子的结构参数 lambda。
- [x, resnorm, residual, exitflag, output, lambda, jacobian] = 99 99lsqcurvefit(...) 返回解 *x* 处的雅可比矩阵。

各调用格式中，fun 变量为目标函数。fun 变量需要输入向量 ***x***，返回 ***x*** 处的目标函数向量 ***F***。可以指定 fun 为一有两个输入参数 ***x*** 和 ***xdata*** 的命令行对象。如

```
f = ...
inline('x(1)*xdata.^2+x(2)*sin(xdata)', 'x', 'xdata');
```

另外，fun 变量可以是一个包含函数名的字符串。该函数可以是 M 文件、内部文件或 MEX 文件。若 fun='myfun'，则该 M 文件应该具有下面的形式：

```
function F = myfun(x,xdata)
F = ...          % 计算 x 处的函数值
```

若雅可比矩阵也可以求得，并且 options.Jacobian 通过下式设置为'on'，

```
options = optimset('Jacobian', 'on')
```

则函数 fun 必须在第 2 个输出变量中输出 *x* 处的雅可比矩阵 ***J***。注意，当被调用的 fun 函数只有一个输出变量（此时优化算法只需要 ***F*** 值而不需要 ***J***），则可以通过核对 nargout 值来避免计算雅可比矩阵。

```
function [F, J] = myfun(x,xdata)
F = ...             % x 处的目标函数值
if nargout > 1      % 两个输出变量
  J = ...           % x 处的雅可比矩阵
end
```

若 fun 函数返回 *m* 个元素的向量（或矩阵）和长度为 *n* 的 *x*，则雅可比矩阵是一个 $m \times n$ 的矩阵，其中 $J(I, j)$为 $F(i)$对 $x(j)$的偏导数。（注意，雅可比矩阵 ***J*** 是梯度 ***F*** 的转换。）

其他参数的意义可参见表 15-7 和表 15-8。

对于不同规模的最小二乘问题，本函数使用不同的算法：

- 大型优化问题——默认时，lsqnonlin 函数将选择大型优化算法。该算法是基于内部映射牛顿法的子空间置信域法。算法的每一次迭代步骤都涉及到用 PCG 法求解大型线性系

统得到的近似解。

- 中型优化算法——lsqnonlin 函数通过将 options.LargeScale 设置为'off'来使用 Levenberg-Marquardt 法进行一维搜索。可通过设置 options.LevenbergMarquardt 参数来选择 Gauss-Newton 法进行一维搜索。设置的具体方法是将 options.LevenbergMarquardt 设置为'off' (且 options.LargeScale 设置为'off')。当残差较小时，使用 Gauss-Newton 法效果更好，速度更快。
- 将 options.LineSearchType 设置为'quadcubic'时，将二次、三次混合插值法作为默认的一维搜索法。将 options.LineSearchType 设置为'cubicpoly'，将采用三次多项式插值法进行一维搜索。该法一般需要较少的函数评价次数，但需要更多的梯度评价次数。这样，如果提供了梯度信息或梯度容易求得，则三次多项式插值是一个更佳的选择。

使用本函数有以下一些限制：

- 进行最小化的函数必须是连续的。lsqcurvefit 函数可能会给出局部最优解。
- lsqnonlin 函数只用于实数变量。当 x 为复数变量时，必须将它分解为实数部分和虚数部分。
- 大型优化问题。lsqnonlin 函数使用大型方法时不求解待定系统：它需要方程数（如，$\boldsymbol{F}$ 的元素个数）至少与变量的个数相等。对于待定系统，将使用中型算法进行求解。如果存在边界条件，将给出警告信息并且将忽略边界条件对问题进行求解。
- 若 x 的元素没有上界或下界，则 lsqnonlin 函数更希望 $\boldsymbol{ub}$ 或 $\boldsymbol{lb}$ 的对应元素设置为 Inf 或–Inf，不希望给 $\boldsymbol{ub}$ 一个很大的正数，而给 $\boldsymbol{lb}$ 一个很小的负数。
- 目前，若在 fun 函数中没有提供解析的雅可比矩阵，选项参数 DerivativeCheck 将不能与大型算法一起使用以比较解析的雅可比矩阵和有限差分得到的雅可比矩阵。取而代之，应该通过将 options 参数 MaxIter 设置为 0 次迭代来使用中型方法核对导数。然后使用大型方法求解问题。
- 中型算法中要求没有边界约束。由于大型算法不能求解待定系统，中型算法不能求解有边界约束的问题，所以具备这两个特征的优化问题都不能用 lsqnonlin 函数进行求解。

22.6.3 应用实例

【例 22-5】 数据向量 ***xdata*** 和 ***ydata*** 的长度为 n。要求找到拟合方程的最佳系数 $\boldsymbol{x}$。

$$\boldsymbol{ydata}(\boldsymbol{i}) = \boldsymbol{x}(1) \bullet \boldsymbol{xdata}(\boldsymbol{i})^2 \bullet \boldsymbol{x}(2) \bullet \sin(\boldsymbol{xdata}(\boldsymbol{i})) + \boldsymbol{x}(3) \bullet \boldsymbol{xdata}(\boldsymbol{i})^3$$

即，希望最小化

$$\min_x \frac{1}{2}\sum_{i=1}^{n}(\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{xdata}_i) - \boldsymbol{ydata}_i)^2$$

式中，F(x, xdata) = x(1)*xdata.^2 + x(2)*sin(xdata) + x(3)*xdata.^3,初值为 x0 = [0.3, 0.4, 0.1]。

首先编写一个 M 文件 opt22_5o.m，返回有 2 个元素的 F 值。

```
function F = myfun(x, xdata)
F = x(1)*xdata.^2 + x(2)*sin(xdata) + x(3)*xdata.^3;
```

然后，调用优化过程：

```
% 假设通过试验得到数据 xdata 和 ydata
xdata = [3.6 7.7 9.3 4.1 8.6 2.8 1.3 7.9 10.0 5.4];
ydata = [16.5 150.6 263.1 24.7 208.5 9.9 2.7 163.9 325.0 54.3];
x0 = [10, 10, 10]                   %初值
```

```
[x,resnorm] = lsqcurvefit(@opt22_5o, x0, xdata, ydata)
```

注意，当 lsqcurvefit 被调用时，假设存在 xdata 和 ydata 并且是相同大小的向量。它们的大小必须是相同的，因为 fun 函数返回的 F 值必须与 ydata 有相同的大小。

经过 33 次函数评价以后，得到问题的解：

```
x =
    0.2269    0.3385    0.3021
% 残差或平方和
resnorm =
    6.2950
```

残差非零，因为本例中数据还有一些噪声（试验误差）。

第 23 章　方 程 求 解

利用 MATLAB 优化工具箱中的函数，可以求解线性方程（组）和非线性方程（组）。

23.1　线性方程（组）的求解

23.1.1　基本原理与算法

利用左除符号“\”可以求解线性方程（组）。对于线性方程组

$$\boldsymbol{Ax}=\boldsymbol{b}$$

有

$$\boldsymbol{x}=\boldsymbol{A}\backslash\boldsymbol{b}$$

如果 $\boldsymbol{A}$ 为平方矩阵，则除了算法不同，$\boldsymbol{A}\backslash\boldsymbol{b}$ 与 inv($\boldsymbol{A}$)*$\boldsymbol{b}$ 大致相当。如果 $\boldsymbol{A}$ 是一个 $n\times n$ 的矩阵，$\boldsymbol{b}$ 是一个具有 n 个元素的列向量或具有多个此类列向量的矩阵，则 $\boldsymbol{X}=\boldsymbol{A}\backslash\boldsymbol{b}$ 为用高斯消元法得到的方程 $\boldsymbol{AX}=\boldsymbol{b}$ 的解。如果 $\boldsymbol{A}$ 奇异，则会给出警告信息。

如果 $\boldsymbol{A}$ 为 $m\times n$ 的矩阵（$m\neq n$），且 $\boldsymbol{b}$ 为具有 m 个元素的列向量或具有多个此类列向量的矩阵，则 $\boldsymbol{X}=\boldsymbol{A}\backslash\boldsymbol{b}$ 为方程组 $\boldsymbol{AX}=\boldsymbol{b}$ 的最小二乘意义上的解。

23.1.2　应用实例

【例 23-1】　求解下列线性方程组

$$\begin{cases} x_1-2x_2+3x_3=2 \\ 3x_1-2x_2+x_3=7 \\ x_1+x_2-x_3=1 \end{cases}$$

该方程组可以写成向量形式

$$\begin{bmatrix} 1 & -2 & 3 \\ 3 & -2 & 1 \\ 1 & 1 & -1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}=\begin{bmatrix} 2 \\ 7 \\ 1 \end{bmatrix}$$

于是可以编程求解：

```
A=[1  -2   3
   3  -2   1
   1   1  -1];
b=[2; 7; 1];
x=A\b
x =
    1.6250
   -1.5000
   -0.8750
```

所以 x_1, x_2 和 x_3 分别为 1.6250、-1.5000 和-0.8750。

【例 23-2】　求解下面的线性方程组：

$$3x_1 + 11x_2 - 2x_3 = 7$$
$$x_1 + x_2 - 2x_3 = 4$$
$$x_1 - x_2 + x_3 = 19$$

问题的组成和求解如下：

```
A = [ 3 11 -2; 1 1 -2; 1 -1 1];
b = [ 7; 4; 19];
x = A\b
x =
   13.2188
   -2.3438
    3.4375
```

23.2　非线性方程（组）的求解

23.2.1　非线性方程的求解

非线性方程的求解问题可以看成是单变量的最小化问题，通过不断缩小搜索区间来逼近问题解的真值。单变量最小化问题的求解方法请参见前面的内容。在 MATLAB 中，非线性方程求解所采用的算法是二分法、secant 法和逆二次内插法的组合。

用 fzero 函数求单变量函数的零点，其调用格式为：

● x = fzero(fun, x0) 如果 x0 为标量，函数试图找到 x0 附近 fun 函数的零点。fzero 函数返回的 x 值为 fun 函数改变符号处邻域内的点，或者是 NaN（如果搜索失败）。这里，当函数发现 Inf、NaN 或复数时，搜索终止。若 x0 为一长度为 2 的向量，fzero 函数假设 x0 为一区间，其中 fun(x0(1))的符号与 fun(x0(2))的符号相反。当该情况不为真时，发生错误。用此区间调用 fzero 函数可以保证 fzero 函数返回 fun 函数改变符号处附近的点。

● x = fzero(fun,x0,options) 用 options 结构指定的优化参数进行最小化。

● x = fzero(fun,x0,options,P1,P2,...) 提供其他变量如 P1, P2 等并传递给目标函数 fun。如果没有选项设置，则令 options=[ ]。

● [x, fval] = fzero(...) 返回解 x 处目标函数的值。

● [x, fval, exitflag] = fzero(...) 返回 exitflag 参数，描述退出条件。

● [x, fval, exitflag, output] = fzero(...) 返回包含优化信息的输出结构 output。

各参数的意义可参见表 15-7 和表 15-8。

fzero 函数是一个 M 文件。其算法由 T. Dekker 发明，是二分法、secant 法和逆二次内插法的组合。

使用 fzero 函数时需要注意的问题有：

➢ 调用 fzero 函数时，使用初值区间（二元素的 x0）常常比用标量 x0 快。
➢ fzero 命令给零点的定义是函数与 X 轴相交的点。函数与 X 轴接触但没有穿过 X 轴的点不算有效零点。例如，y = x.^2 函数曲线便在 0 处与 X 轴接触，但没有穿过 X 轴，所以没有发现零点。对于没有有效零点的函数，fzero 函数将一直运行到发现 Inf，NaN 或复数值。

【例 23-3】 通过计算 sin 函数在 3 附近的零点来计算π：

```
x = fzero(@sin,3)
x =
   3.1416
```

【例 23-4】 找出下面函数的零点：

$$f(x) = x^3 - 2x - 5$$

首先编写目标函数的 M 文件 opt23_4o.m。

```
function y = f(x)
y = x.^3-2*x-5;
```

寻找 2 附近的零点：

```
z = fzero(@opt23_4o,2)
z =
   2.0946
```

因为本函数为一多项式，故函数 roots([1 0 -2 -5])也能找到相同的实数零点和共轭复数解。

```
2.0946
-1.0473 +1.1359i
-1.0473 -1.1359I
```

23.2.2 非线性方程组的求解

非线性方程组的数学模型为

$$\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$$

式中，$\boldsymbol{x}$ 为一向量，$\boldsymbol{F}(\boldsymbol{x})$为一函数，返回向量值。

在 MATLAB 中，用 fsolve 函数求解非线性方程组。其算法基于最小二乘法。fsolve 函数的调用格式为：

- x = fsolve(fun, x0) 初值为 x0，试图求解由 fun 函数描述的等式系统。
- x = fsolve(fun, x0, options) 用 options 结构指定的优化参数进行最小化。
- x = fsolve(fun, x0, options, P1, P2,...) 将问题参数 P1、P2 等直接传递给 fun 函数。
- [x, fval] = fsolve(fun, x0) 返回解 x 处的目标函数 fun 的值。
- [x, fval, exitflag] = fsolve(...) 返回 exitflag 参数，描述函数的退出条件。
- [x, fval, exitflag, output] = fsolve(...) 返回包含优化信息的结构输出 output。
- [x, fval, exitflag, output, jacobian] = fsolve(...) 返回解 x 处的 fun 函数的雅可比（Jacob）矩阵。

各调用格式中，fun 变量为目标函数，即需要最小化的目标函数。fun 函数需要输入标量参数 x，返回 x 处的目标函数标量值 f。可以将 fun 函数指定为命令行，如

```
x = fminbnd(inline('sin(x*x)'),x0)
```

同样，fun 参数可以是一个包含函数名的字符串。对应的函数可以是 M 文件、内部函数或 MEX 文件。若 fun='myfun'，则 M 文件函数 myfun.m 必须有下面的形式。

```
function F = myfun(x)
F = ...          % 计算 x 处的目标函数
```

若雅可比矩阵可以算得，并且 options.Jacobian 设置为'on',即

```
options = optimset('Jacobian','on')
```

则 fun 函数必须在第二个输出变量中输出 x 处的雅可比矩阵 $\boldsymbol{J}$。注意，如果 fun 函数只要求有一个输出变量（此时优化算法只要求目标函数值而不要求雅可比值），则通过核对 nargout

参数可以避免计算雅可比矩阵。

```
function [F,J] = myfun(x)
F = ...            % x 处的目标函数值
if nargout > 1     % 两个输出变量
  J = ...          % x 处的雅可比矩阵
end
```

若 fun 函数返回一个有 m 个元素的向量或矩阵，长度为 n 的 x，则雅可比矩阵 $\boldsymbol{J}$ 为 $m\times n$ 的矩阵，其中 $J(i, j)$为 $F(i)$对 $x(j)$的偏导数。

options 变量为优化参数选项。可以用 optimset 函数设置或改变这些选项。有些参数适用于所有的算法，有的只适用于大型算法，而另外一些则只适用于中型算法。

首先描述适用于大型问题的选项。这仅仅是一个参考，因为使用大型问题算法有一些条件。对于 fminunc 函数来说，必须提供梯度信息。对于 LargeScale 选项，当设为'on'时使用大型算法，若设为'off'则使用中型问题的算法。

适用于大型和中型算法的参数包括：

- Diagnostics——打印最小化函数的诊断信息。
- Display——显示水平。选择'off'，不显示输出；选择'iter'，显示每一步迭代过程的输出；选择'final'，显示最终结果。打印最小化函数的诊断信息。
- Jacobian——用户定义的目标函数的雅可比矩阵。
- MaxFunEvals——函数评价的最大次数。
- MaxIter——最大允许迭代次数。
- TolFun——函数值的终止容限。
- TolX——处的终止容限。

只用于大型算法的参数包括：

- JacobPattern——用于有限差分的 Jacob 矩阵的稀疏形式。若不便求 fun 函数的稀疏 Jacob 矩阵 $\boldsymbol{J}$，可以通过用梯度的有限差分获得的 $\boldsymbol{J}$ 的稀疏结构（如非零值的位置等）来得到近似的 Jacob 矩阵 $\boldsymbol{J}$。若连矩阵的稀疏结构都不知道，则可以将 JacobPattern 设为密集矩阵，在每一次迭代过程中，都将进行密集矩阵的有限差分近似（这是默认设置）。这将非常麻烦，所以花一些力气得到 Jacob 矩阵的稀疏结构还是值得的。
- MaxPCGIter——PCG 迭代的最大次数。
- PrecondBandWidth——PCG 前处理的上带宽，默认时为零。对于有些问题，增加带宽可以减少迭代次数。
- TolPCG——PCG 迭代的终止容限。
- TypicalX——典型 x 值。
- 只用于中型算法的参数：
- DerivativeCheck——对用户提供的导数和有限差分求出的导数进行对比。
- DiffMaxChange——变量有限差分梯度的最大变化。
- DiffMinChange——变量有限差分梯度的最小变化。
- LineSearchType——一维搜索算法的选择。

其他参数的意义可参见表 15-7 和表 15-8。

非线性方程组求解的算法基于非线性最小二乘法，方程组在 lsqnonlin 函数中也使用了该

算法。使用最小二乘法的好处是如果方程组没有零解，该算法仍然返回一个残差很小的点。但是，若系统的雅可比矩阵是奇异矩阵，则该算法将会收敛于一点，该点不是方程组的解。根据问题规模的不同，选择不同的求解算法。

① 大型优化问题。默认时，fsolve 函数会选择大型算法。该算法是基于内部映射牛顿法的子空间置信域法，每一次迭代都与 PCG 法求解大型线性方程组得到的近似解有关。

② 中型优化问题。fsolve 函数将 options.LargeScale 设置为'off'，使用 Gauss-Newton 法进行一维搜索。也可以选择 Levenberg-Marquardt 法进行一维搜索。该算法可通过设置 options.LevenbergMarquardt 来实现。将 options.LevenbergMarquardt 设置为'on'（并将 options.LargeScale 设置为'off')来选择 Levenberg-Marquardt 法。

一维搜索算法根据选项确定，如将 options.LineSearchType 设置为'quadcubic'，则默认的一维搜索算法为二次、三次混合插值法。将 options.LineSearchType 设置为'cubicpoly'，将采用三次多项式插值法进行一维搜索。该法一般需要较少的函数评价次数，但需要更多的梯度评价次数。这样，如果提供了梯度信息或梯度容易求得，则三次多项式插值是一个更佳的选择。

注意，fsolve 函数可能会收敛于非零点并给出下面的信息：

Optimizer is stuck at a minimum that is not a root Try again with a new starting guess

此时，重新给定初值并运行 fsolve 函数。

另外，使用 fsolve 函数时还有下面一些要求：

- 要求解的函数必须是连续的。当成功收敛时，fsolve 函数只给出一个根。当函数收敛于非零点时，用其他初值进行试算。
- fsolve 函数只对实数变量有效。当 x 为复数时，必须将它分为实数部分和虚数部分。
- 若在 fun 函数中提供了雅可比矩阵，options 参数不能与大型算法同时使用以比较解析雅可比矩阵和有限差分雅可比矩阵。可以通过将参数 MaxIter 设置为 0 来核对导数。然后用大型算法重新运行程序。

【例 23-5】 求解下列方程组：

$$2x_1 - x_2 = e^{-x_2}$$
$$-x_1 + 2x_2 = e^{-x_2}$$

假设初值为 $\boldsymbol{x0}$ = [-5, -5]。

首先编写一个 M 文件 opt23_5o.m，计算 $\boldsymbol{x}$ 处等式的值 $\boldsymbol{F}$。

```
function F = myfun(x)
F = [2*x(1) - x(2) - exp(-x(1));
     -x(1) + 2*x(2) - exp(-x(2))];
```

下一步调用优化函数：

```
x0 = [-5; -5];                           % 初值
options=optimset('Display', 'iter');     % 输出显示的选项
[x, fval] = fsolve(@opt23_5o, x0, options) % 调用优化函数
```

经过 28 次函数评价以后，得到表 23-1 所示的结果。

表 23-1 计算结果

ITERATION	FUNC-COUNT	F(X)	NORM OF STEP	FIRST-ORDER OPTIMALITY	CG-ITERATIONS
1	4	47071.2	1	2.29e+004	0
2	7	6527.47	1.45207	3.09e+003	1

续表

ITERATION	FUNC-COUNT	F(X)	NORM OF STEP	FIRST-ORDER OPTIMALITY	CG-ITERATIONS
3	10	918.372	1.49186	418	1
4	13	127.74	1.55326	57.3	1
5	16	14.9153	1.57591	8.26	1
6	19	0.779051	1.27662	1.14	1
7	22	0.00372453	0.484658	0.0683	1
8	25	9.21617e-008	0.0385552	0.000336	1
9	28	5.66133e-017	0.000193707	8.34e-009	1

表中各列的意义分别为：

ITERATION——迭代次数；

FUN-COUNT——目标函数计算次数；

F(X)——目标函数值；

NORM OF STEP——当前步长的范数；

FIRST-ORDER OPTIMILITY——当前梯度的无限范数；

CG-ITERATIONS——当前迭代步中源于 PCG 的迭代次数。

```
Optimization terminated successfully:
Relative function value changing by less than OPTIONS.TolFun
x =
    0.5671
    0.5671
fval =
    1.0e-08*
   -0.5320
   -0.5320
```

说明优化过程成功终止，函数值的相对改变小于 OPTIONS.TolFun。x_1 和 x_2 的取值均为 0.5671，两个函数的目标值均为 1.0e-08*-0.5320。

【例 23-6】 求矩阵 $\boldsymbol{x}$，使其满足方程

$$\boldsymbol{X} * \boldsymbol{X} * \boldsymbol{X} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

初值为 $\boldsymbol{x} = [1, 1; 1, 1]$

首先编写一待求等式的 M 文件 opt23_6o.m。

```
function F = myfun(x)
F = x*x*x-[1, 2; 3, 4];
```

然后调用优化过程：

```
x0 = ones(2, 2);                % 设初值
options = optimset('Display', 'off');  % 取消显示
[x, Fval, exitflag] = fsolve(@opt23_6o, x0, options)
```

解为：

```
x =
    -0.1291   0.8602
     1.2903   1.1612
```

```
Fval =
    1.0e-03*
    0.1541   -0.1163
    0.0109   -0.0243
exitflag =
    1
```

残差接近于零。

```
sum(sum(Fval.*Fval))
ans =
    3.7974e-008
```

第 24 章　大 型 课 题

24.1　概述

随着工程技术的发展和最优化技术的广泛应用，科学技术领域中需要求解很多大规模问题。大型问题的解法近年来在国内外颇受重视，并做了很多工作，主要是将拟 Newton 法推广到大规模问题。在求解变量个数相当大的大型问题时，遇到的最大困难是需要非常大的存储量。例如，Newton 法、变尺度法为了计算逐次搜索方向需要存储 n 阶矩阵，这样就受到计算机存储量的限制。而且，由此引起的计算时间过长。

但是，实际遇到的大规模问题一般都有两个十分有利的性质，即

（1）很多大型无约束问题，由于某些变量之间没有非线性的交互作用，故它的 Hess 矩阵含有大量的零元素。

（2）大规模问题中出现的矩阵往往具有一定的结构或模式。在一个具有一定结构的矩阵中，零或非零元素不是随机分布的，根据问题的固有性质和变量之间的关系，对它们在矩阵中所占的位置事先有所了解。对于稀疏的大规模问题，通常利用 Hess 矩阵的稀疏性和结构性进行求解，设法减少计算量和存储量（例如，与零元素的乘法可以不做；在有特定结构的矩阵中，大块的零元素可以不存储等），把 Newton 法，特别是拟 Newton 法推广到大型问题。

对于非稀疏的大规模问题，目前最有效的算法还是共轭梯度法和一些以它和拟 Newton 法为基础的修正算法。

表 24-1 描述了 MATLAB 中能求解的大型问题的数学模型和对应函数。在前面介绍各优化问题的优化函数时，我们也介绍了对应大型问题的算法。读者朋友们可以在解决具体问题时参考理解和使用。

表 24-1　大型问题模型与函数表

函　　数	问 题 模 型	附 加 条 件	备　　注
Fminunc	$\min\limits_{x} f(\boldsymbol{x})$	必须提供 $f(\boldsymbol{x})$ 的梯度	提供 Hess 矩阵的稀疏结构，或计算目标函数的 Hess 矩阵 Hess 矩阵必须是稀疏的
Fmincon	$\min\limits_{x} f(\boldsymbol{x})$　　$\boldsymbol{l} \leqslant \boldsymbol{x} \leqslant \boldsymbol{u}(\boldsymbol{l} < \boldsymbol{u})$ $\min\limits_{x} f(\boldsymbol{x})$ $\boldsymbol{Aeq} \cdot \boldsymbol{x} = \boldsymbol{beq}$ $\boldsymbol{Aeq}$ 为 m×n 矩阵(m≤n)	必须提供 $f(\boldsymbol{x})$ 的梯度	提供 Hess 矩阵的稀疏结构，或计算目标函数的 Hess 矩阵 Hess 矩阵必须是稀疏的
Lsqnonlin	$\min\limits_{x} \frac{1}{2}\lVert \boldsymbol{F}(\boldsymbol{x}) \rVert_2^2 = \frac{1}{2}\sum\limits_{i} \boldsymbol{F}_i(\boldsymbol{x})^2$ $\min\limits_{x} \frac{1}{2}\lVert \boldsymbol{F}(\boldsymbol{x}) \rVert_2^2 = \frac{1}{2}\sum\limits_{i} \boldsymbol{F}_i(\boldsymbol{x})^2$　　$\boldsymbol{l} \leqslant \boldsymbol{x} \leqslant \boldsymbol{u}(\boldsymbol{l} < \boldsymbol{u})$ $\boldsymbol{F}(\boldsymbol{x})$ 的方程个数必须与变量个数相等	不可供使用	提供雅可比矩阵的稀疏结构，或计算目标函数的雅可比矩阵 雅可比矩阵必须是稀疏的

续表

函　　数	问 题 模 型	附 加 条 件	备　　注
Lsqcurvefit	$\min\limits_{x}\frac{1}{2}\lVert \boldsymbol{F}(\boldsymbol{x},\boldsymbol{xdata})-\boldsymbol{ydata}\rVert_2^2$ $\min\limits_{x}\frac{1}{2}\lVert \boldsymbol{F}(\boldsymbol{x},\boldsymbol{xdata})-\boldsymbol{ydata}\rVert_2^2$　　$\boldsymbol{l}\leqslant\boldsymbol{x}\leqslant\boldsymbol{u}(\boldsymbol{l}<\boldsymbol{u})$	不可供使用	提供雅可比矩阵的稀疏结构，或计算目标函数的雅可比矩阵 雅可比矩阵必须是稀疏的
Fsolve	$\boldsymbol{F}(\boldsymbol{x})=0$	不可供使用	
lsqlin	$\min\limits_{x}\lVert \boldsymbol{C}\bullet\boldsymbol{x}-\boldsymbol{a}\rVert_2^2$　　$\boldsymbol{l}\leqslant\boldsymbol{x}\leqslant\boldsymbol{u}(\boldsymbol{l}<\boldsymbol{u})$ $\boldsymbol{C}$ 为 $m\times n$ 矩阵（$m\geqslant n$）		$\boldsymbol{C}$ 必须是稀疏矩阵
Linprog	$\min\limits_{x}\boldsymbol{f}^T\boldsymbol{x}$ $\boldsymbol{A}\bullet\boldsymbol{x}\leqslant\boldsymbol{b}$ $\boldsymbol{Aeq}\bullet\boldsymbol{x}=\boldsymbol{beq}$ $\boldsymbol{l}\leqslant\boldsymbol{x}\leqslant\boldsymbol{u}$	不可供使用	$\boldsymbol{A}$ 和 $\boldsymbol{Aeq}$ 必须是稀疏矩阵
quadprog	$\min\limits_{x}\frac{1}{2}\boldsymbol{x}^T\boldsymbol{Hx}+\boldsymbol{f}^T\boldsymbol{x}$　　$\boldsymbol{l}\leqslant\boldsymbol{x}\leqslant\boldsymbol{u}(\boldsymbol{l}<\boldsymbol{u})$ $\min\limits_{x}\frac{1}{2}\boldsymbol{x}^T\boldsymbol{Hx}+\boldsymbol{f}^T\boldsymbol{x}$ $\boldsymbol{Aeq}\bullet\boldsymbol{x}=\boldsymbol{beq}$ $\boldsymbol{Aeq}$ 为 $m\times n$ 矩阵($m\leqslant n$)	不可供使用	$\boldsymbol{H}$ 和 $\boldsymbol{Aeq}$ 必须是稀疏矩阵

24.2　带雅可比矩阵的非线性等式

对于下面的问题，寻找 $\boldsymbol{x}$，使 $f(\boldsymbol{x})=0$，假设 $n=1000$。

$$F(\boldsymbol{x})=3x_1-2x_1^2-2x_2+1$$

$$\cdots\cdots$$

$$F(i)=3x_i-2x_i^2-x_{i-1}-2x_{i+1}+1$$

$$\cdots\cdots$$

$$F(n)=3x_n-2x_n^2-x_{n-1}+1$$

用 fsolve 函数求解大型非线性方程组。

第 1 步：建立 M 文件 nlsf1.m，计算目标函数值和雅可比矩阵。

```
Function  [F,J]=nlsf1(x);
%评价向量函数
n=length(x);
F=zeros(n,1);
i=2:(n-1);
F(i)=(3-2*x(i)).*x(i)-x(i-1)-2*x(i+1)+1;
F(n)=(3-2*x(n)).*x(n)-x(n+1)+1;
F(1)=(3-2*x(1)).*x(1)-2*x(2)+1;
%如果 vargout>0，则评价雅可比矩阵
if nargout>1
  d=-4*x+3*ones(n, 1);
  D=sparse(1: n, 1:n,d, n, n);
  C=-2*ones(n-1, 1);
  E=sparse(2: n, 1: n-1, e, n, n);
```

```
  J=C+D+E;
end
```

第 2 步：调用等式求解过程系统。

```
xstart=ones(1000,1);
fun= @nlsf1;
options=optimset('Display', 'iter', 'Jacobian', 'on');
[x, fval, exitflag, output]=fsolve(fun, xstart, options);
```

初始点与函数名一起给出。默认时使用大型算法，所以不必在 options 变量中进行指定。还指定了每一次迭代的输出。最后，fsolve 函数将使用 nlsf1.m 中可以得到的雅可比矩阵信息，但需要将雅可比矩阵参数用 optimset 函数设置为'on'。

上面的命令得到以下结果：

Iteration	Func-count	f(x)	Norm of step	First-order optimality	CG-Iterations
1	2	1011	1	19	0
2	3	16.1942	7.91898	2.35	3
3	4	0.0228027	1.33142	0.291	3
4	5	0.000103359	0.0433329	0.0201	4
5	6	7.3792e-007	0.0022606	0.000946	4
6	7	4.02299e-010	0.000268381	4.12e-005	5

表中各列的意义分别为：

- Iteration——第几次迭代；
- Func-count——函数计算次数；
- f(x)——目标函数的值；
- Norm of steps——当前步的范数；
- First-order optimality——当前梯度的无限范数；
- CG-Iterations——当前迭代步中源于 PCG 的迭代次数。

优化过程成功收敛：

- 函数值的相对改变值小于 OPTIONS.TolFun。
- 在每个主要的迭代步，使用预设结合梯度法对线性系统进行了求解。在 options 参数中，PrecondBandWidth 的默认值取为 0。
- 从一阶优化值可以看出，过程达到了快速收敛。每一次主要迭代所需要的结合梯度（CG）迭代次数较低，对于大小为 1000 的问题，最多只需要 5 次，表示在本问题中，该线性系统不是很难求解的。

在使用参数 PrecondBandWidth 的整个过程中，可以通过选择带宽预设值来代替对角线上带宽预设值的默认选择。如果需要使用三对角预设值，则将 PrecondBandWidth 参数设为 1。

```
options = optimset('Display','iter','Jacobian','on',...
                   'PrecondBandWidth',1) ;
[x,fval,exitflag,output] = fsolve(fun,xstart,options) ;
```

本例中，输出为：

Iteration	Func-count	f(x)	Norm of step	First-order optimality	CG-Iterations
1	2	1011	1	19	0
2	3	16.0839	7.92496	1.92	1

3	4	0.0458181	1.3279	0.579	1
4	5	0.000101184	0.0631898	0.0203	2
5	6	3.16615e-007	0.00273698	0.00079	2
6	7	9.72481e-010	0.00018111	5.82e-005	2

优化成功终止：

- 函数值的相对改变小于 OPTIONS.TolFun。
- 尽管迭代次数相同，但 PCG 迭代的次数下降了，所以每一次迭代中的工作量更小。

24.3 采用梯度和 Hess 矩阵的非线性最小化

假设有下面的问题

$$f(\boldsymbol{x})=\sum_{j=1}^{n-1}[(\boldsymbol{x}_j^2)^{x+1}+(\boldsymbol{x}_{j+1}^2)^{(x+1)}]$$

式中，n=1000。这是一个大型优化问题，现在求解它。

第 1 步：建立计算目标函数和目标函数导数、Hessian 矩阵的三对角稀疏矩阵。

```
type brownfgh
% 计算函数值
  n=length(x);
  y=zeros(n,1);
  i=1:(n-1);
  y(i)=(x(i).^2).^(x(i+1).^2+1)+(x(i+1).^2).^(x(i).^2+1);
  f=sum(y);
%
% 计算梯度值
  if nargout > 1
     i=1:(n-1); g = zeros(n,1);
     g(i)= 2*(x(i+1).^2+1).*x(i).*((x(i).^2).^(x(i+1).^2))+...
           2*x(i).*((x(i+1).^2).^(x(i).^2+1)).*log(x(i+1).^2);
     g(i+1)=g(i+1)+...
             2*x(i+1).*((x(i).^2).^(x(i+1).^2+1)).*log(x(i).^2)+...
             2*(x(i).^2+1).*x(i+1).*((x(i+1).^2).^(x(i).^2));
  end
%
% 计算 Hess 矩阵
  if nargout > 2
     v=zeros(n,1);
     i=1:(n-1);
     v(i)=2*(x(i+1).^2+1).*((x(i).^2).^(x(i+1).^2))+...
           4*(x(i+1).^2+1).*(x(i+1).^2).*(x(i).^2).*((x(i).^2).^ ...
          ((x(i+1).^2)
-1))+2*((x(i+1).^2).^(x(i).^2+1)).*(log(x(i+1).^2));
     v(i)=v(i)+4*(x(i).^2).*((x(i+1).^2).^(x(i).^2+1)).*…
          ((log(x(i+1).^2)).^2);
     v(i+1)=v(i+1)+2*(x(i).^2).^(x(i+1).^2+1).*(log(x(i).^2))+...
           4*(x(i+1).^2).*((x(i).^2).^(x(i+1).^2+1)).* ...
           ((log(x(i).^2)).^2)+2*(x(i).^2+1).*((x(i+1).^2).^(x(i).^2));
```

```
v(i+1)=v(i+1)+4*(x(i).^2+1).*(x(i+1).^2).*(x(i).^2).*((x(i+1).^2).^…
        (x(i).^2-1));
    v0=v;
    v=zeros(n-1,1);
    v(i)=4*x(i+1).*x(i).*((x(i).^2).^(x(i+1).^2))+...
          4*x(i+1).*(x(i+1).^2+1).*x(i).*((x(i).^2).^(x(i+1).^2)).*…
          log(x(i).^2);
    v(i)=v(i)+
4*x(i+1).*x(i).*((x(i+1).^2).^(x(i).^2)).*log(x(i+1).^2);
    v(i)=v(i)+4*x(i).*((x(i+1).^2).^(x(i).^2)).*x(i+1);
    v1=v;
    i=[(1:n)';(1:(n-1))'];
    j=[(1:n)';(2:n)'];
    s=[v0;2*v1];
    H=sparse(i,j,s,n,n);
    H=(H+H')/2;
  end
```

第 2 步：调用给定初值 xstart 的非线性最小化过程。

```
n=1000;
xstart=-ones(n,1);
xstart(2:2:n,1)=1;
options=optimset('GradObj','on','Hessian','on');
   [x,fval,exitflag,output]=fminunc(@brownfgh,xstart,options);
x =…          （略）
fval =
  2.8709e-017
exitflag =
     1
output =
      iterations: 8
       funcCount: 8
    cgiterations: 7
   firstorderopt: 4.7948e-010
       algorithm: 'large-scale: trust-region Newton'
```

24.4　采用梯度和 Hess 稀疏模式的非线性最小化

现在解决与上一节相同的问题，但 Hess 矩阵通过有限差分近似得到。使用 fminunc 函数求解大型问题，首先必须计算目标函数的梯度（对于中型问题，就没有这个必要）。

第 1 步：建立计算目标函数和其导数的 M 文件 brownfg.m。

```
function [f,g]=brownfg(x,dummy)
n=length(x);
y=zeros(n,1);
i=1:(n-1);
y(i)=(x(i).^2).^(x(i+1).^2+1)+…
    (x(i+1).^2).^(x(i).^2+1);
f=sum(y);
if nargout>1
```

```
    i=1:(n-1);
    g=zeros(n,1);
    g(i)=2*(x(i+1).^2+1).*x(i).*…
        ((x(i).^2).^(x(i+1).^2))+…
        2*x(i).*((x(i+1).^2).^(x(i).^2+1)).*…
        log(x(i+1).^2);
    g(i+1)=g(i+1)+…
        2*x(i+1).*((x(i).^2).^(x(i+1).^2+1)).*…
        log(x(i).^2)+…
        2*(x(i).^2+1).*x(i+1).*…
        ((x(i+1).^2).^(x(i).^2));
end
```

为了使 Hess 矩阵 ***H***(***x***)的稀疏有限差分近似计算更加有效，必须首先确定稀疏结构 *H*。在本例中，假设稀疏矩阵 Hstr 可以从文件 brownhstr.mat 中得到。使用 spy 命令，会发现 Hstr 确实是稀疏的（只有 2998 个非零值）。用 optimset 函数将 HessPattern 参数设置为 Hstr。当与本问题大小相当的问题具有明显的稀疏结构时，不设置 HessPattern 参数将需要很大一部分不必要的内存和计算，因为 fminunc 函数将试图对一个有百万非零入口的完全 Hess 矩阵进行有限差分。

还需要用 optimset 参数将 GradObj 参数设置为'on'，因为梯度在 brownfg.m 文件中进行计算。fminunc 函数的运行过程在第 2 步显示。

第 2 步：调用给定初值 xstart 的非线性最小化过程

```
fun= @brownfg;
load brownhstr    % 获得 Hess 矩阵的结构 Hstr
Spy(Hstr)         % 查看 Hstr 的稀疏结构
n=1000;
xstart=-ones(n,1);
xstart(2:2:n,1)=1;
options=optimset('GradObj','on','HessPattern',Hstr);
[x,fval,exitflag,output]=fminunc(fun,xstart,options);
```

生成如图 24-1 所示的稀疏结构图。

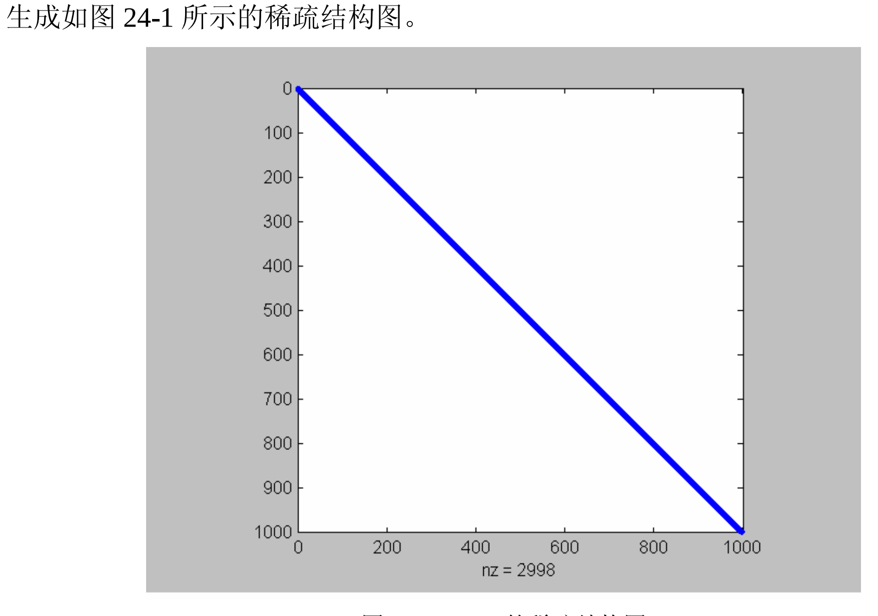


图 24-1　Hstr 的稀疏结构图

这个具有 1000 个变量的大型问题通过 8 次迭代和 7 次结合梯度迭代达到收敛（exitflag 参数为正表示收敛）。解 x 处的最终函数值和优化度量都十分接近于 0（对于 fminunc 函数，一阶优化为函数梯度的有限范数，它是局部极小点）。

```
exitflag =
     1
fval =
  7.4738e-017
output.iterations
ans =
     8
output.cgiterations
ans =
     7
output.firstorderopt
ans =
     7.9822e-010
```

24.5 给定边界约束和初始条件的非线性最小化

假设有下面的非线性最小化问题：

$$f(\boldsymbol{x}) = 1 + \sum_{j=1}^{k} \left|(3-2x_j)x_j - x_{j-1} + 1\right|^p + \sum_{j=1}^{n/2} \left|x_j + x_{j+n/2}\right|^p$$

$$-10 \leqslant x_j \leqslant 10$$

$$n = 800$$

$$p = 7/3$$

$$x_0 = x_{n+1} = 0$$

按照下面的步骤求解它。

第 1 步：建立计算目标函数和目标函数的导数的 M 文件 tbroyfg.m。

```
n=length(x);  % n 应该是 4 的倍数
p=7/3;
y=zeros(n,1);
i=2:(n-1);
y(i)= abs((3-2*x(i)).*x(i)-x(i-1)-x(i+1)+1).^p;
y(n)= abs((3-2*x(n)).*x(n)-x(n-1)+1).^p;
y(1)= abs((3-2*x(1)).*x(1)-x(2)+1).^p;
j=1:(n/2);
z=zeros(length(j),1);
z(j)=abs(x(j)+x(j+n/2)).^p;
f=1+sum(y)+sum(z);
%
% 计算梯度
if nargout > 1
   p=7/3; n=length(x); g = zeros(n,1); t = zeros(n,1);
   i=2:(n-1);
   t(i)=(3-2*x(i)).*x(i)-x(i-1)-x(i+1)+1;
```

```
    g(i)= p*abs(t(i)).^(p-1).*sign(t(i)).*(3-4*x(i));
    g(i-1)=g(i-1)-p*abs(t(i)).^(p-1).*sign(t(i));
    g(i+1)=g(i+1)-p*abs(t(i)).^(p-1).*sign(t(i));
    tt = (3-2*x(n)).*x(n)-x(n-1)+1;
    g(n)=g(n)+p*abs(tt).^(p-1).*sign(tt).*(3-4*x(n));
    g(n-1)=g(n-1)-p*abs(tt).^(p-1).*sign(tt);
    tt=(3-2*x(1)).*x(1)-x(2)+1;
    g(1)=g(1)+p*abs(tt).^(p-1).*sign(tt).*(3-4*x(1));
    g(2)=g(2)-p*abs(tt).^(p-1).*sign(tt);
    j=1:(n/2); t(j)=x(j)+x(j+n/2);
    g(j) = g(j)+p*abs(t(j)).^(p-1).*sign(t(j));
    jj=j+(n/2);
    g(jj) = g(jj)+p*abs(t(j)).^(p-1).*sign(t(j));
    grad = g;
end
```

Hess 矩阵的稀疏模式已经确定并保存在文件 tbroyhstr.mat 中。本问题中 Hess 矩阵的稀疏结构可以利用 spy 命令绘出。

```
load tbroyhstr
spy(Hstr)
```

生成的图形如图 24-2 所示。图中，中心条带为一五带宽的矩阵，用下面的语句生成的图形（图 24-3）显示得更清楚。

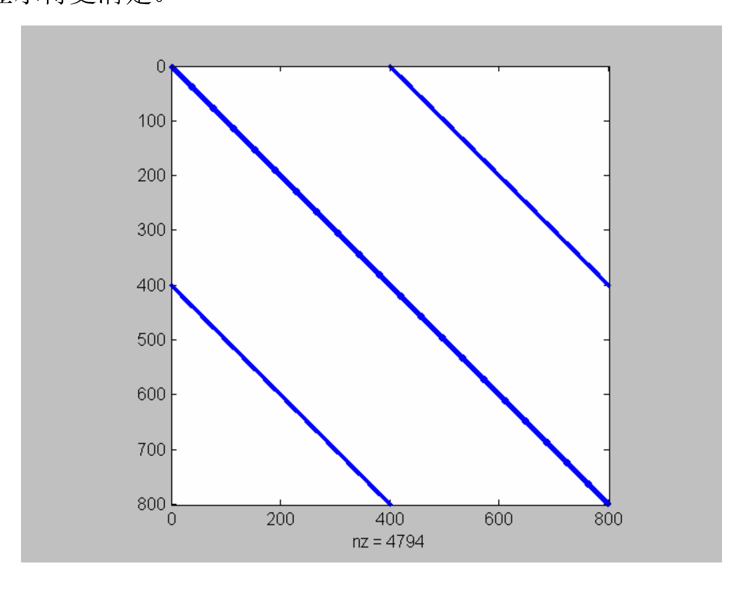


图 24-2　本问题 Hess 矩阵的稀疏结构图

```
spy(Hstr(1:20,1:20))
```

用 optimset 函数将 HessPattern 参数设置为 Hstr。与本问题大小相同并具有明显稀疏结构的问题，如果不设置 HessPattern 参数，将需要很大的不必要的内存和计算。因为 fmincon 函数将试图对一个具有 640000 个非零入口的完全 Hess 矩阵使用有限差分法。

还必须使用 optimset 函数将 GradObj 参数设置为'on'，因为在 tbroyfg.m 中计算了梯度。fmincon 函数的具体运行见第 2 步。

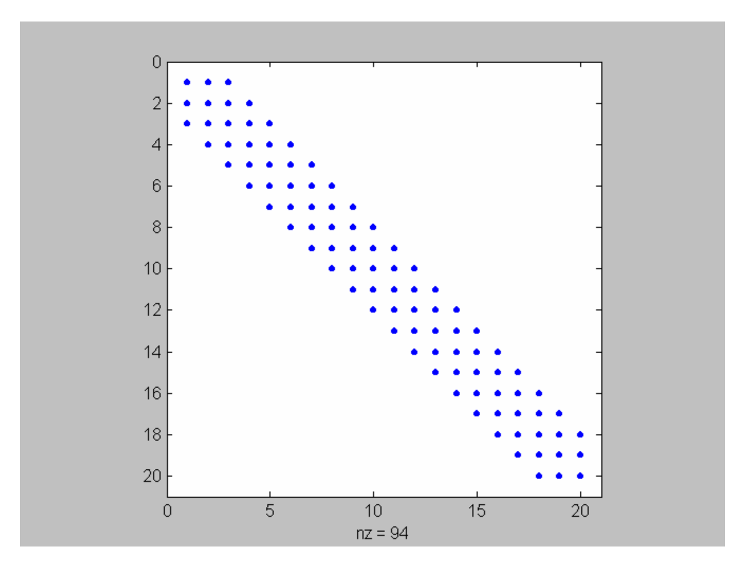


图 24-3　中心条带为五带宽矩阵

第 2 步：调用给定初值的非线性最小化过程

```
fun= @tbroyfg;
load tbroyhstr     % 求 Hess 矩阵的稀疏结构
n=800;
xstart=-ones(n,1);
xstart(2:2:n)=1;
lb=-10*ones(n,1);
ub=-lb;
options=optimset('GradObj','on','HessPattern',hstr);
[x,fval,exitflag,output]=…
   fmincon('tbroyfg',xstart,[],[],[],[],lb,ub,[],options);
```

经过 8 次迭代以后，输出结果：

```
exitflag =
    1
fval =
  270.4790
output =
       iterations: 8
        funcCount: 8
     cgiterations: 18
    firstorderopt: 0.0163
        algorithm: 'large-scale: trust-region reflective Newton'
```

对于边界约束问题，一阶优化为 v.*g 的有限范数，其中 v 定义为 Box 约束，g 为梯度。

因为是五带宽中心条带，可以通过用五带宽预设值代替默认的对角预设值来改进求解过程。使用 optimset 函数，重新将 PrecondBandWidth 参数设置为 2 并重新求解问题（带宽为上对角或下对角的个数，不包含主对角）。

```
fun = @tbroyfg;
load tbroyhstr         % 得到 Hess 结构 Hstr。
n = 800;
```

```
xstart = -ones(n, 1); xstart(2: 2: n, 1) = 1;
lb = -10*ones(n, 1); ub = -lb;
options = optimset('GradObj', 'on', 'HessPattern', Hstr, ...
                'PrecondBandWidth',2);
[x, fval, exitflag, output] = ...
fmincon(fun, xstart, [ ], [ ], [ ], [ ], lb, ub, [ ], options);
```

迭代次数增加了两次，但总的 CG 迭代次数从 18 降到了 15。

```
exitflag =
    1
fval =
    2.7048e+002
output =
      iterations: 10
       funcCount: 10
    cgiterations: 15
   firstorderopt: 7.5339e-005
       algorithm: 'large-scale: trust-region reflective Newton'
```

24.6　带等式约束的非线性最小化

如果没有其他约束存在，则用 fmincon 函数求解大型问题可以控制等式约束。假设现在希望最小化的目标函数与上一节的相同，其代码包含在函数 brownfgh.m 中，其中 $n = 1000$，对于 ***Aeq***，有 100 个方程（所以 ***Aeq*** 是一个 100×1000 的矩阵）。

第 1 步：建立计算目标函数和目标函数梯度、Hess 矩阵的 M 文件 brownfgH.m

```
function [f, g, H] = brownfgH(x)
% 计算函数值
  n=length(x);
  y=zeros(n, 1);
  i=1:(n-1);
  y(i)=(x(i).^2).^(x(i+1).^2+1)+(x(i+1).^2).^(x(i).^2+1);
  f=sum(y);
%
% 计算梯度值
  if nargout > 1
     i=1:(n-1); g = zeros(n,1);
     g(i)= 2*(x(i+1).^2+1).*x(i).*((x(i).^2).^(x(i+1).^2))+...
             2*x(i).*((x(i+1).^2).^(x(i).^2+1)).*log(x(i+1).^2);
     g(i+1)=g(i+1)+...
              2*x(i+1).*((x(i).^2).^(x(i+1).^2+1)).*log(x(i).^2)+...
              2*(x(i).^2+1).*x(i+1).*((x(i+1).^2).^(x(i).^2));
  end
%
% 计算 Hess 矩阵
  if nargout > 2
     v=zeros(n,1);
     i=1:(n-1);
```

```
    v(i)=2*(x(i+1).^2+1).*((x(i).^2).^(x(i+1).^2))+...
          4*(x(i+1).^2+1).*(x(i+1).^2).*(x(i).^2).*((x(i).^2).^ ...

((x(i+1).^2)-1))+2*((x(i+1).^2).^(x(i).^2+1)).*(log(x(i+1).^2));
    v(i)=v(i)+4*(x(i).^2).*((x(i+1).^2).^(x(i).^2+1)).*…
          ((log(x(i+1).^2)).^2);
    v(i+1)=v(i+1)+...
            2*(x(i).^2).^(x(i+1).^2+1).*(log(x(i).^2))+...
            4*(x(i+1).^2).*((x(i).^2).^(x(i+1).^2+1)).* ...
   j=[(1:n)';(2:n)'];
   s=[v0;2*v1];
    H=sparse(i,j,s,n,n);
((log(x(i).^2)).^2)+2*(x(i).^2+1).*((x(i+1).^2).^(x(i).^2));

v(i+1)=v(i+1)+4*(x(i).^2+1).*(x(i+1).^2).*(x(i).^2).*((x(i+1).^2).^…
           (x(i).^2-1));
    v0=v;
    v=zeros(n-1,1);
    v(i)=4*x(i+1).*x(i).*((x(i).^2).^(x(i+1).^2))+...
          4*x(i+1).*(x(i+1).^2+1).*x(i).*((x(i).^2).^(x(i+1).^2)).*…
          log(x(i).^2);
    v(i)=v(i)+
4*x(i+1).*x(i).*((x(i+1).^2).^(x(i).^2)).*log(x(i+1).^2);
    v(i)=v(i)+4*x(i).*((x(i+1).^2).^(x(i).^2)).*x(i+1);
    v1=v;
    i=[(1:n)';(1:(n-1))'];

    H=(H+H')/2;
  end

     load browneq
     condest(Aeq*Aeq’)
     ans=
        2.9310e+006
```

因为 brownfgh 函数计算目标函数值、梯度值和 Hess 矩阵值，所以需要使用 optimset 函数表示利用 GradObj 参数和 Hess 参数在 brownfgh 函数中可以得到该信息。

在文件 browneq.mat 中可以得到稀疏矩阵 ***Aeq*** 和向量 ***beq***。

```
load browneq
```

线性约束系统是 100×1000 的。

```
condest(Aeq*Aeq')
ans =
   2.9310e+006
```

第 2 步：调用非线性过程，给定初值。

```
fun = @brownfgh;
load browneq
n=1000;
```

```
xstart=-ones(n, 1);
xstart(2:2:n)=1;
options=optimset('GradObj', 'on', 'Hessian', 'on', 'PrecondBandWidth',
inf);
[x, fval, exitflag, output]=fmincon('brownfgh', xstart, [ ], [ ], Aeq, beq,
[ ],…
    [ ], [ ], options)
```

将参数 PrecondBandWidth 设置为 inf，将用稀疏直接求解器代替预设结合梯度法。

exitflag 参数表示经过 16 次迭代以后过程收敛。

```
exitflag =
    1
fval =
    205.9313
output =
      iterations: 16
       funcCount: 16
    cgiterations: 14
   firstorderopt: 2.1434e-004
       algorithm: 'large-scale: projected trust-region Newton'
```

在 x 处，线性等式得到满足。

```
norm(Aeq*x-beq)
ans =
   1.1913e-012
```

24.7　带边界约束的二次最小化

最小化带上界和下界的二次大型问题，可以使用 quadprog 函数。本节要讨论的问题保存在 MAT 文件 qpbox1.mat 中，是一个正定二次问题，Hess 矩阵 ***H*** 为三对角阵，有上界（***ub***）和下界（***lb***）。

下面装载 Hess 矩阵，定义 *f*、***lb*** 和 ***ub***，调用给定初值 xstart 的二次最小化过程。

```
load qbbox1     % 装载 H
lb=zeros(400, 1);
lb(400)=-inf;
ub=0.9*ones(400, 1);
ub(400)=inf;
f=zeros(400, 1);
f([1 400])=-2;
xstart=0.5*ones(400, 1);
[x, fval, exitflag,output]=quadprog(H, f, [ ], [ ], [ ], [ ], lb, ub, xstart);
```

计算结果为：

```
exitflag =
    1
output =
   firstorderopt: 7.8435e-006
      iterations: 20
```

```
       cgiterations: 1809
          algorithm: 'large-scale: reflective trust-region'
```

经过 20 次迭代计算收敛，较高的 CG 迭代次数表示线性求解的损耗是比较大的。改变设置以后进行计算：

```
options = optimset('MaxPCGIter', 50);
[x,fval,exitflag,output] = ...
        quadprog(H, f, [ ], [ ], [ ], [ ], lb, ub, xstart, options);
```

现在计算仍然收敛，但 CG 迭代次数减少了（1547 次）。

```
exitflag =
     1
output =
    firstorderopt: 2.3821e-005
       iterations: 36
     cgiterations: 1547
        algorithm: 'large-scale: reflective trust-region'
```

第 2 个方案是通过将 PrecondBandWidth 参数设置为 inf 来使用直接求解器进行计算。

```
options = optimset('PrecondBandWidth', inf);
[x, fval, exitflag, output] = ...
        quadprog(H, f, [ ], [ ], [ ], [ ], lb, ub, xstart, options);
```

现在迭代次数减少到 10 次。

```
exitflag =
     1
output =
    firstorderopt: 4.8955e-007
       iterations: 10
     cgiterations: 9
        algorithm: 'large-scale: reflective trust-region'
```

在每个迭代步使用直接求解法一般可以大幅度地减少迭代次数，但每次迭代时所花的时间要多一些。

24.8　带边界约束的线性最小二乘问题

许多情况下都要遇到稀疏线性二乘问题，变量具有边界约束。下一个问题要求变量非负。本问题源于对线性样条进行函数拟合。特殊情况下，许多散点散布在单位方形中，需要近似的函数对这些点进行评价，在线性系数非负的情况下建立起分段线性样条近似。本例中，用 2000 个方程拟合 400 个变量。

```
load particle   % 获取 C, d
lb = zeros(400, 1);
[x,resnorm,residual,exitflag,output] = ...
            lsqlin(C, d, [], [], [], [], lb);
exitflag =
     1
resnorm =
    22.5794
```

```
output =
      algorithm: 'large-scale: trust-region reflective Newton'
   firstorderopt: 2.7870e-005
     iterations: 10
    cgiterations: 42
```

对于边界约束问题，一阶优化是 v.*g 的有限范数，其中 v 定义为 Box 约束，g 为梯度。

在每个迭代过程中用系数 QR 因子分解可以改进一阶优化：将 PrecondBandWidth 设置为 inf。

```
options = optimset('PrecondBandWidth', inf);
[x, resnorm, residual, exitflag, output] = ...
            lsqlin(C, d, [ ], [ ], [ ], [ ], lb, [ ], [ ], options);
```

迭代次数和一阶优化都减小了。

```
exitflag =
    1
resnorm =
   22.5794
output =
      algorithm: 'large-scale: trust-region reflective Newton'
   firstorderopt: 5.5907e-015
     iterations: 12
    cgiterations: 11
```

24.9 带等式约束和不等式约束的线性规划问题

该问题的数学模型为

$$
\begin{aligned}
&\min \boldsymbol{f}^T \boldsymbol{x} \\
&\boldsymbol{Aeq} \cdot \boldsymbol{x} = \boldsymbol{beq} \\
&\boldsymbol{A} \cdot \boldsymbol{x} \leqslant \boldsymbol{b} \\
&\boldsymbol{x} \geqslant 0
\end{aligned}
$$

可以键入下面的命令，在 MATLAB 工作空间装入矩阵和向量 ***A***, ***Aeq***, ***b***, ***beq***, ***f*** 和下界 ***lb***。

load sc50b

在 sc50b.mat 中，本问题有 48 个变量、30 个不等式约束和 20 个等式约束。

可以使用 linprog 函数求解此问题：

```
[x, fval, exitflag, output] = ...
   linprog(f, A, b, Aeq, beq, lb, [ ], [ ], optimset('Display', 'iter'));
```

因为设置 optimset 函数时，要求显示迭代信息，所以输出到命令行窗口的结果为：

```
Residuals:   Primal     Dual     Duality    Total
             Infeas    Infeas      Gap       Rel
             A*x-b    A'*y+z-f    x'*z      Error
------------------------------------------------------
Iter    0:  1.50e+003  2.19e+001  1.91e+004  1.00e+002
Iter    1:  1.15e+002  2.94e-015  3.62e+003  9.90e-001
Iter    2:  1.16e-012  2.21e-015  4.32e+002  9.48e-001
```

```
Iter   3:  3.23e-012  5.16e-015  7.78e+001  6.88e-001
Iter   4:  5.78e-011  7.61e-016  2.38e+001  2.69e-001
Iter   5:  9.31e-011  1.84e-015  5.05e+000  6.89e-002
Iter   6:  2.96e-011  1.62e-016  1.64e-001  2.34e-003
Iter   7:  1.51e-011  2.74e-016  1.09e-005  1.55e-007
Iter   8:  1.51e-012  2.37e-016  1.09e-011  1.51e-013
```

优化过程成功终止。

exitflag 的值为正，说明 linprog 函数收敛。此外，还可以得到最终函数值 fval 和迭代次数 output.iterations。

```
exitflag =
     1
fval =
    -70.0000
output =
      iterations: 8
    cgiterations: 0
       algorithm: 'lipsol'
```

第三篇　偏微分方程数值解工具箱

第 25 章　偏微分方程数值解工具箱概述

工程中有许多问题可以归结为偏微分方程问题，如弹塑性力学中研究对象（结构、边坡等）内部的应力应变问题、地下水渗流问题等。这些由偏微分方程及边界条件、初始条件等组合成的数学模型，只有在十分特殊的条件下才能求得解析解。因此，在很长一段时间内，人们对于这一类问题是无能为力的。随着计算机技术的发展，各种数值方法应运而生，如有限元法、有限差分法、离散元法、拉格朗日元法等等。利用数值法，可以求得这些问题的数值解。它不是问题的精确解，但可以无限接近精确解。MATLAB 采用有限元法求解偏微分方程的数值解。

有限元法由我国数学家冯康等于 1956 年最先提出，最早应用于结构力学。它是利用剖分插值把区域连续求解的微分方程离散成求解线性代数方程组，以近似解代替精确解。按照所依据的原理不同，有限元法又可分为变分有限元法、迦辽金有限元法和均衡有限元法等。

在进行有限元计算以前，往往要进行一系列的预处理工作，如对该几何形状或形体进行离散化，即用比较简单的形状或形体来逼近和代替实际的形状或形体。例如，对于二维的情况，可以用很多三角形或四边形来逼近一个有着复杂边界的实际图形；对于三维的情况，可以用四面体或六面体的集合体来逼近实际形体。这样，可以将比较复杂的曲线或曲面问题转化为相对简单的直线或平面问题。这个过程，即为建立几何模型的过程。图 25-1 中用三角形网格将研究域离散化。

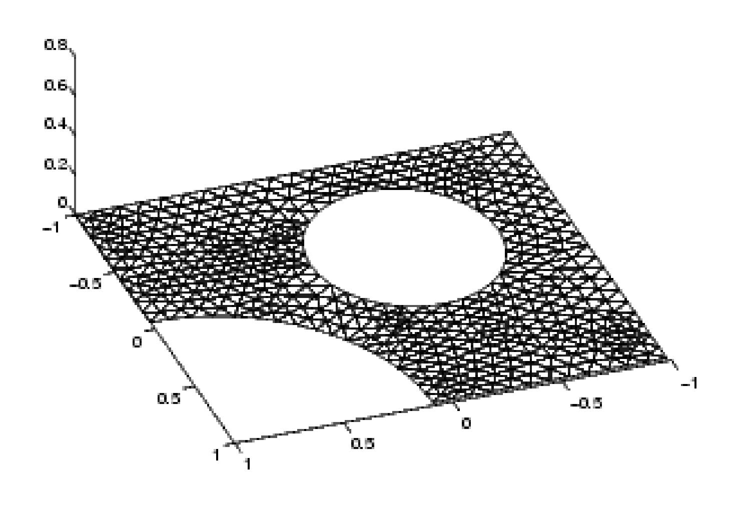


图 25-1　研究域的三角形网格

PDE 工具箱提供了一系列函数绘制研究对象的几何形状并根据几何形状自动进行网格剖分。表 25-1 中为绘制几何形状的函数，表 25-2 中为与网格剖分有关的函数。

表 25-1　绘图算法函数

函　　数	描　　述
pdecirc	绘圆
pdeellip	绘椭圆
pdemdlcv	将 PDE 工具箱 1.0 模型的 M 文件转换为 PDE 工具箱 1.0.2 版本的格式
pdepoly	绘多边形
pderect	绘矩形
pdetool	PDE 工具箱图形用户集成界面（GUI）

表 25-2　几何算法函数

函　　数	描　　述
csgchk	核对几何描述矩阵的有效性
csgdel	删除最小子域之间的界线
decsg	将建设性实体几何模型分解为最小子域
initmesh	创建初始三角形网格
jigglemesh	微调三角形网格的内部点
pdearcl	在参数表示和圆弧长度之间进行内插
poimesh	在矩形几何图形上生成规则网格
refinemesh	将一个三角形网格细化
wbound	写边界条件指定文件
wgeom	写几何指定函数

建立几何模型以后，需要根据要求解的问题赋予初始条件、边界条件和方程参数等。组合各方面的贡献以后，组成一个大型的矩阵方程，然后采用一定的方法求得方程的解。表 25-3 中列出了 PDE 工具箱提供的部分偏微分方程求解算法函数。

表 25-3　偏微分方程求解算法函数

函　　数	描　　述
adaptmesh	生成自适应网格并求解 PDE 问题
assema	组合面积的整体贡献
assemb	组合边界条件的贡献
assempde	组合刚度矩阵和 PDE 问题的右端项
hyperbolic	求解双曲线 PDE 问题
parabolic	求解抛物线型 PDE 问题
pdeeig	求解特征值 PDE 问题
pdenonlin	求解非线性 PDE 问题
poisolv	在矩形网格上对泊松方程进行快速求解

计算完毕以后，可以得到网格上各个节点（网格顶点）上的解，可以用列表的形式来列出这些值，但使用图形更直观。于是引出有限元计算的后处理问题。通常用于表现有限元计算结果的图形有变形网格图、云图（色谱图）、等值线图、矢量图、网格图、表面图、流线图等。PDE 工具箱提供的后处理函数如表 25-4 所示。

表 25-4　绘图函数

函　　数	描　　述
pdecont	绘等值线图
pdegplot	绘制 PDE 几何图
pdemesh	绘 PDE 三角形网格
pdeplot	一般 PDE 工具箱绘图函数
pdesurf	绘三维表面图

第 26 章　偏微分方程数值解有关函数介绍

26.1　偏微分方程求解算法函数

1．adaptmesh 函数

利用该函数生成适应性网格并进行 PDE 求解。其调用格式和说明如下所示：

● [uz, p, e, t]=adaptmesh(g, b, c, a, f, 'PropertyName', PropertyValue,) 进行适应性网格生成和 PDE 求解。选项参数作为属性名称/属性值匹配对给出。

该函数生成椭圆型标量 PDE 问题的解

$$-\nabla \cdot (c\nabla u) + au = f \qquad \text{在}\Omega\text{上}\ ,$$

或椭圆型系统 PDE 问题的解

$$-\nabla \cdot (c\otimes\nabla u) + au = f \qquad \text{在}\Omega\text{上}$$

问题的几何条件和边界条件由 g 和 b 给出。网格由 p、e 和 t 进行描述。

解 u 代表解向量 ***u***。

求解 PDE 问题序列的算法用到三角形网格。第 1 个三角形网格的生成是通过给 adaptmesh 函数赋选项参数或通过调用不带参数的 initmesh 函数得到的。下面生成的网格是通过求解 PDE 问题、计算误差估计、选择一系列基于误差估计的三角形并最后定义这些网格来获得的。然后重新计算 PDE 问题。此循环直到用三角形选择方法获得网格或达到最大三角形数或要求生成的三角形已经全部生成为止。

g 描述 PDE 问题的分解几何模型。g 可以是分解几何矩阵或几何模型 M 文件的文件名。分解几何矩阵和几何模型 M 文件的格式分别在 decsg 和 pdegeom 中描述。

b 描述 PDE 问题的边界条件。b 可以是边界条件矩阵或边界 M 文件的文件名。边界条件矩阵和边界 M 文件的格式分别在 assemb 和 pdebound 中描述。

PDE 问题的适应性三角形网格由网格数据 p、e 和 t 给出。网格数据代表意义的详情请参见 initmesh 函数。

PDE 问题的系数 c，a 和 f 可以通过多种方式给出。若非线性求解器通过设置属性 nonlin 得到了加强的话，这些系数可以通过 u 求出。系数与时间 t 无关。在 assempde 函数中可以看到所有选项的一个完整的列表。

表 26-1 列出了属性名/属性值匹配对，及它们的默认值和属性描述。

Par 被传递给 Tripick 函数。（Tripick 函数将在后面描述。）它通常作为描述解与方程拟合程度的容限值。

当达到 Ngen 或网格中的三角形个数超过 Maxt 时，细化（加密）终止。

p1、e1 和 t1 为输入网格数据。该三角形网格被作为适应性算法的初始网格。initmesh 函数中给出了网格数据的细节。如果没有提供初始网格，将用调用不带参数的 initmesh 函数得到的结果作为初始网格。

表 26-1　属性表

属　性　名	属　性　值	默　认　值	属 性 描 述
Maxt	positive integer	inf	新三角形的最大个数
Ngen	positive integer	10	生成三角形的最大次数
Mesh	p1, e1, t1	initmesh	初始网格
Tripick	MATLAB function	pdeadworst	三角形选择方法
Par	numeric	0.5	函数参数
Rmethod	longest|regular	longest	三角形细化方法
Nonlin	on|off	off	使用非线性求解器
Toln	numeric	1e-4	非线性容限
Init	u0	0	非线性初值
Jac	fixed|lumped|full	fixed	非线性雅可比矩阵的计算
Norm	numeric|inf|energy	inf	非线性残差范数

三角形选择方法 Tripick 是一种用户定义的三角形选择方法。给定通过 pdejmp 函数计算的误差估计以后，将选择在下一次网格生成中将要细化的网格。使用变量 *p*，*t*，*cc*，*aa*，*ff*，*u*，errf 和 par 调用函数。*p* 和 *t* 代表当前生成的三角形，*cc*，*aa* 和 *ff* 为 PDE 问题的当前系数并扩展到三角形的中点，*u* 是当前解，errf 为计算误差估计，par 为函数参数，赋给 adaptmesh 函数作为可选变量。矩阵 *cc*，*aa*，*ff* 和 errf 都有 *Nt* 列,其中 *Nt* 为三角形的当前个数。*cc*，*aa* 和 *ff* 中的行数与输入变量 *c*，*a* 和 *f* 的相同。errf 中的每一行都对应系统中的某一个方程。工具箱中有两个标准的三角形选择方法—pdeadworst 和 pdeadgsc。pdeadworst 选择 errf 超过最坏值阀值（默认值为 0.5）的三角形，pdeadgsc 则使用相对容限临界值来选择三角形。

网格的细化方法有最长法和匀称法。细化方法的细节内容可参见 refinemesh 函数。

适应性算法也可以求解非线性 PDE 问题。对于非线性 PDE 问题，必须将 Nonlin 参数设置为 on。非线性容限 Toln,非线性初值 u0,非线性雅可比矩阵 Jac 和非线性残差范数 Norm 将被传递到非线性求解器 pdenonlin 中去。参见 pdenonlin 函数，可了解非线性求解器的细节内容。

【例 26-1】　求解扇形圆弧的拉普拉斯方程。该扇形圆弧的 Dirichlet 边界条件为：沿圆弧 $u = \cos(2/3\mathrm{atan2}(y, x))$，沿直线 $u = 0$。并与其解析解进行对比。

首先用最坏误差临界值法细化网格，直到至少有 500 个三角形：

```
[u, p, e, t]=adaptmesh('cirsg', 'cirsb', 1, 0, 0, 'maxt', 500,...
                      'tripick', 'pdeadworst', 'ngen', inf);
x=p(1,:); y=p(2,:);
exact=((x.^2+y.^2).(1/3).*cos(2/3*atan2(y, x)))';
max(abs(u-exact))
ans =
   0.0058
size(t, 2)
ans =
   534
pdemesh(p,e,t)
```

结果如图 26-1 所示。最大绝对误差为 0.0058，有 534 个三角形。

下面我们使用均匀三角形网络进行细化：

```
[p, e, t]=initmesh('cirsg');
[p, e, t]=refinemesh('cirsg', p, e, t);
u=assempde('cirsb', p, e, t, 1, 0, 0);
x=p(1,:); y=p(2,:);
exact=((x.^2+y.^2).^(1/3).*cos(2/3*atan2(y, x)))';
max(abs(u-exact))
ans =
      0.0085
size(t, 2)
ans =
      1640
[p, e, t]=refinemesh('cirsg', p, e, t);
u=assempde('cirsb', p, e, t, 1, 0, 0);
x=p(1,:); y=p(2,:);
exact=((x.^2+y.^2).^(1/3).*cos(2/3*atan2(y, x)))';
max(abs(u-exact))
ans =
      0.0054
size(t, 2)
ans =
      6560
pdemesh(p, e, t)
```

细化后的网格如图 26-2 所示。可见，使用均匀细化法，需要 6560 个三角形就可以达到比用适应法更好的绝对误差。对于有正则解的问题，我们希望误差为 $o(h^2)$。但本问题中解是奇异的，因为在原点上 $u \approx r1/3$。

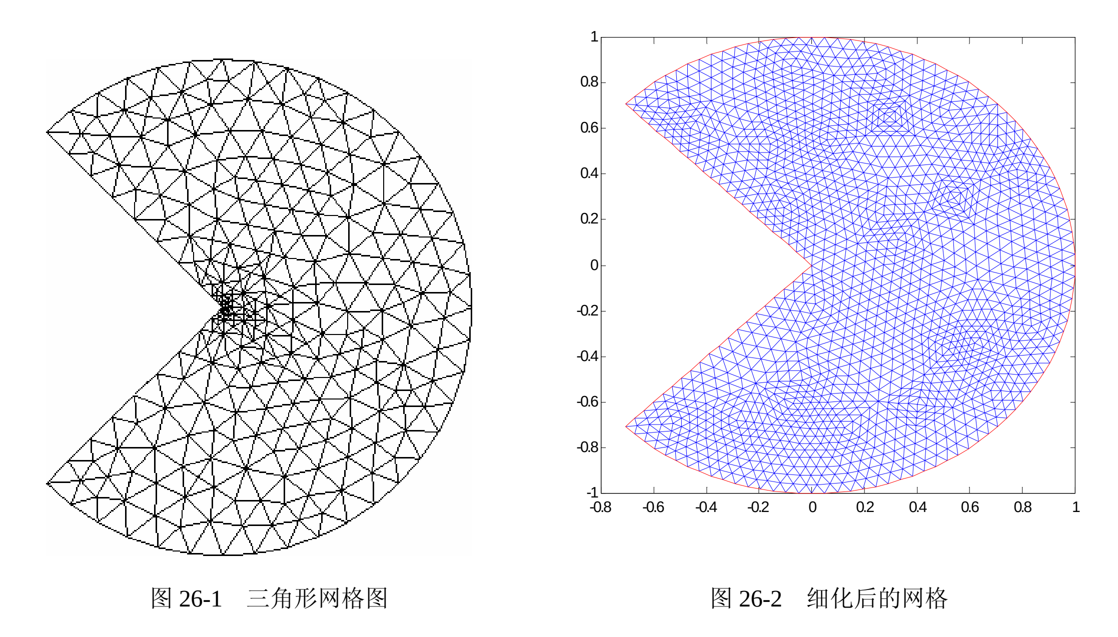


图 26-1　三角形网格图　　　　图 26-2　细化后的网格

上面的计算终止时，会给出下面消息中的一种：

Adaption completed（这表示 Tripick 返回 0 值，即细化三角形的个数为 0）。

Maximum number of triangles obtained（达到三角形最大个数）。

Maximum number of refinement passes obtained（得到的最大细化次数）。

2．assema 函数

该函数组合面积的整体贡献，其调用格式为：

● [K, M, F]=assema(p, t, c, a, f)　组合刚度矩阵 K、总体矩阵 M 和右侧向量 F。输入参数 p, t, c, a, f, u0, time 和 sdl 的意义与后面 assempde 函数中的相同。

3．assemb 函数

该函数组合边界条件的贡献，其调用格式为：

● [Q, G, H, R]=assemb(b, p, e)　组合矩阵 Q 和 H,以及向量 G 和 R。Q 应该被添加到系统矩阵中去，并且包含混合边界条件的贡献。G 需要添加到右端项中并包含广义 Neumann 条件和混合边界条件。方程 H*u=R 代表 Dirichlet 类型的边界条件。输入参数 p，e，u0，time 和 sdl 具有与 assempde 函数中相同的含义。b 描述 PDE 问题的边界条件。b 可以是边界条件矩阵或边界 M 文件的文件名。边界条件矩阵的格式用下面的形式进行描述，边界 M 文件在 pdebound 函数中进行描述。

PDE 工具箱处理下列边界条件类型：

（1）在广义 Neumann 边界段，q 和 g 与根据下式得到的导数有关：

$$\vec{\boldsymbol{n}}\cdot(c\nabla u)+qu=g$$

（2）在 Dirichlet 边界段，$hu=r$。

该工具箱也处理研究区域Ω上的偏微分方程组。令方程组的变量个数为 N，边界条件为

$$hu=r$$

$$\vec{\boldsymbol{n}}\cdot(c\otimes\nabla u)+\underline{q}u=g+h'\mu$$

$\vec{\boldsymbol{n}}\cdot(c\otimes\nabla u)$为 N*1 的矩阵，它的（$i$-1）元素为

$$\sum_{j=1}^{N}\left(\cos(\alpha)\,c_{i,j,1,1}\frac{\partial}{\partial x}+\cos(\alpha)c_{i,j,1,2}\frac{\partial}{\partial y}+\sin(\alpha)c_{i,j,2,1}\frac{\partial}{\partial x}+\sin(\alpha)c_{i,j,2,2}\frac{\partial}{\partial y}\right)u_j$$

式中，α 为边界法向向量的向量角，指向区域Ω外侧。

在 PDE 图形用户界面中，边界条件矩阵由系统自动创建，并由 assemb 函数将边界的贡献传递给矩阵 $\boldsymbol{Q}$、$\boldsymbol{G}$、$\boldsymbol{H}$ 和 $\boldsymbol{R}$。边界条件矩阵也可以作为一个 M 文件保存起来，并与 wbound 函数一起使用。

对于分解几何矩阵中的每一列而言，必须在边界条件矩阵中有对应的列。每一列的格式可查看下表：

第 1 行包含系统的维数 N。

第 2 行包含 Dirichlet 边界条件的个数 M。

第 3 行至第 $3+N^2-1$ 行包含代表 $\underline{\boldsymbol{q}}$ 的字符串的长度，各长度按照列的大小顺序保存。

第 $3+N^2$ 行到第 $3+N^2+N-1$ 行包含代表 g 的字符串的长度。

第 $3+N^2+N$ 行至第 $3+N^2+N+MN-1$ 行包含代表 h 的字符串的长度。长度根据 h 的列的大小顺序进行保存。

第 $3+N^2+N+NM$ 行至第 $3+N^2+N+MN+M-1$ 行包含代表 r 的字符串的长度。

接下来的各行所包含的 MATLAB 文本代表实际的边界条件函数。文本字符串的长度根据上面的内容决定。MATLAB 文本根据矩阵 $\boldsymbol{h}$ 和 $\underline{\boldsymbol{q}}$ 的列的大小顺序进行保存。字符串之间没有

分隔符。可以插入包含下列变量的 MATLAB 表达式：

- 二维坐标 x，y。
- 边界段参数 s，它与圆弧的长度成比例。在边界段的起点处 s 等于 0，并沿箭头指向的方向在边界段上增加到 1。
- 向外的法向向量元素 nx 和 ny。如果需要切向向量，则可以通过 nx 和 ny 得到，因为 tx = -ny，ty = nx。
- 解 u（只有当指定了输入变量 u 以后才有），时间 t（只有在输入变量中指定了才有）。

【例 26-2】 描述边界条件矩阵的格式。对于标量 PDE（N=1）问题的一条 Neumann 边界（M=0），

$$\vec{n} \cdot (c\nabla u) = -x^2$$

边界条件将由列向量来代表，即

```
[1 0 1 5 '0' '-x.^2']'
```

注意，h 或 r 中没有保存长度。

同样，对于标量 PDE 问题，Dirichlet 边界条件

$$u = x^2 - y^2$$

也保存在列向量中。

```
[1 1 1 1 1 9 '0' '0' '1' 'x.^2-y.^2']'
```

对于 N=2 且混合边界条件（M=1）如下所示的系统：

$$(h_{11} h_{12})u = r_1$$

$$\vec{n} \bullet (\underline{c} \otimes \nabla u) + \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix} u = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} + s$$

其列可看做：

```
2
1
lq11
lq21
lq12
lq22
lg1
lg2
lh11
lh12
lr1
q11 ...
q21 ...
q12 ...
q22 ...
g1 ...
g2 ...
h11 ...
h12 ...
r1 ...
```

其中，lq11, lq21, . . . 表示 MATLAB 文本表达的长度，q11，q21, . . . 表示实际表达。

4．assempde 函数

该函数组合刚度矩阵和 PDE 问题的右端项，其调用格式为：

● assempde 函数是 PDE 工具箱中的一个基本函数。它使用有限元法来组合 PDE 问题。assempde 命令组合标量 PDE 问题。

$$-\nabla\cdot(c\nabla \boldsymbol{u}) + \boldsymbol{au} = \boldsymbol{f} \quad 在\Omega上$$

或系统 PDE 问题

$$-\nabla\cdot(\boldsymbol{c}\otimes\nabla \boldsymbol{u}) + \boldsymbol{au} = \boldsymbol{f} \quad 在\Omega上$$

该命令可以有选择地生成 PDE 问题的解。

对于标量的情况，用解的列向量来代表解向量 u，列向量中的解的值对应于 p 的对应节点。对于有 Np 个节点的 N 维系统，u 的第一个 np 值描述 u 的第一个元素，接下来的 np 值描述 u 的第二个元素，如此类推。这样，u 的元素就作为 N 个节点值块放置到向量 u 中。

● u=assempde(b,p,e,t,c,a,f) 通过在线性方程组剔除 Dirichlet 边界条件来组合和求解 PDE 问题。

● [K,F]=assempde(b,p,e,t,c,a,f) 通过用刚度对 Dirichlet 边界条件近似来组合 PDE 问题。K 和 F 分别为刚度矩阵和右端项。PDE 问题的有限元解为 $u=K\backslash F$。

● [K,F,B,ud]=assempde(b,p,e,t,c,a,f) 通过从线性等式组剔除 Dirichlet 边界条件来组合 PDE 问题。$u1=K\backslash F$ 返回非 Dirichlet 点上的解。完整的 PDE 问题可以通过 MATLAB 表达式 u=B*u1+ud 进行求解。

● [K,M,F,Q,G,H,R]=assempde(b,p,e,t,c,a,f) 给出 PDE 问题的裂区表示(split representation)。

● u=assempde(K,M,F,Q,G,H,R) 将 PDE 问题的裂区表示转化为单一矩阵或向量的形式，然后通过从线性等式组中剔除 Dirichlet 边界条件来进行求解。

● [K1,F1]=assempde(K,M,F,Q,G,H,R) 通过用很大的常数修改 Dirichlet 边界条件来将 PDE 问题的裂区表示转化为单一矩阵或向量的形式。

● [K1,F1,B,ud]=assempde(K,M,F,Q,G,H,R) 通过从线性等式组中剔除 Dirichlet 边界条件来将 PDE 问题的裂区表示转化为单一矩阵或向量形式。

b 描述 PDE 问题的边界条件。b 可以是边界条件矩阵或边界 M 文件的文件名。边界条件和边界 M 文件的格式分别描述在 assemb 函数和 pdebound 函数中。

PDE 问题的几何模型通过网格数据 p，e 和 t 给出。initmesh 函数中给出了网格数据意义的细节。

subdomain 标签的选项列表 sdl 限制列表中标签所标示的子域的组合过程。选项输入变量 u0 和时间分别用于非线性求解器和时间逐步算法。解的初值 u0 的形式与 u 的相同。

标量 PDE 问题中的系数 c, a 和 f 可以通过下面的方式用 MATLAB 变量 c, a 和 f 表示。

（1）常数。

（2）三角形中心点上的值行向量。

（3）三角形中心点上计算系数的 MATLAB 文本表达。该表达式通过上下文进行评价，其中，变量 x, y, sd, u, ux, uy 和 t 为代表三角形中心点处值的行向量（t 为标量）。行向量包括 x, y 坐标、子域标签、解、解对 x 和 y 的偏导数以及时间。只有在 u_0 已经被传递给 assempde 以后 u, u_x 和 u_y 才可以使用。这种情况对于标量 t 也适用，标量 t 作为时间传递给 assempde 函数。

（4）MATLAB 文本表达序列用感叹号作间隔，即 marks !。每一个文本表达的语法必须根据上面的内容确定。序列中表达式的数目必须与三角形列表 t 中子域标签的数目相同（该数

目可以通过键入 max(t(4,:))进行核对）。

（5）自定义的 MATLAB 函数，它接受变量(p, t, u, time)。若对应参数没有传递给 assempde 函数，则 u 和时间为空矩阵。p 和 t 为网格数据，u 为输入参数 u0，time 为时间输入参数。若时间为 NaN（空值）而函数与时间有关，则函数必须返回对应大小的矩阵，矩阵中所有元素均为 NaN。

若 $\boldsymbol{c}$ 包含两个根据上面内容得到的数据行，则它们是 $c_{1,1}$ 和 $c_{2,2}$,是一个 2×2 的对角矩阵的元素。

若 $\boldsymbol{c}$ 有 4 行，它们是 $c_{1,1}, c_{2,1}, c_{1,2}$ 和 $c_{2,2}$，是一个 2×2 的矩阵中的元素。

$$\begin{bmatrix} c_{1,1} & 0 \\ 0 & c_{2,2} \end{bmatrix}$$

令 N 为 PDE 系统的维数。现在 $\boldsymbol{c}$ 是一个 $N \times N \times 2 \times 2$ 的张量，$\boldsymbol{a}$ 是一个 $N \times N$ 的矩阵，$\boldsymbol{f}$ 是一个长度为 N 的列向量。$\boldsymbol{c}$，$\boldsymbol{a}$，$\boldsymbol{d}$ 和 $\boldsymbol{f}$ 的元素 c_{ijkl}，a_{ij}，d_{ij} 和 f_i 按照行的大小保存在 MATLAB 矩阵 $\boldsymbol{c}$，$\boldsymbol{a}$，$\boldsymbol{d}$ 和 $\boldsymbol{f}$ 中。这些矩阵中的每一行在语法上与标量情况是近似的。有一点不同的是：在 MATLAB 文本表达的评价上，变量 $\boldsymbol{u}$，$\boldsymbol{u_x}$ 和 $\boldsymbol{u_y}$ 为包含有 N 行的矩阵，每一行对应一个元素。恒等矩阵、对角矩阵和对称矩阵作为特例处理。对于张量 c_{ijkl}，它适用于下标 i，j，k 和 l。

$\boldsymbol{f}$ 中的行数决定系统的维数 N，$\boldsymbol{f}$ 中的第 i 行代表 f 中的元素 f_i。

根据表 26-2，$\boldsymbol{a}$ 的行数 n_a 与 $\boldsymbol{a}$ 的元素 a_{ij} 有关。对于对称的情况，假设 $j \geqslant i$。所有不能组成的元素都假设为 0：

表 26-2　行数 n_a 与元素的关系

n_a	对称性	a_{ij}	在 $\boldsymbol{a}$ 中的行
1	无	a_{ii}	1
N	无	a_{ij}	I
$N(N+1)/2$	有	a_{ij}	$j(j-1)/2+i$
N^2	无	a_{ij}	$N(j-1)+i$

后面有一个例子演示了 $\boldsymbol{a}$ 是如何保存到 $\boldsymbol{a}$ 矩阵中去的。

$\boldsymbol{c}$ 中的元素决定于维数 N 和 $\boldsymbol{c}$ 中的行数 n_c。n_c 与表 26-3 中第 1 列的 N 相匹配，即按顺序从第 1 行到最后一行。

第 1 次匹配决定 $\boldsymbol{c}$ 的取值类型。对于一些小值($2 \leqslant N \leqslant 4$)，它们的取值仅决定于进行检验的次序。对于对称的情况，假设 $j \geqslant i$，且 $i \geqslant k$。所有不能组成的元素都假设为 0：

表 26-3　行数 n_c 与元素的关系

n_c	对 称 性	c_{ijkl}	在 $\boldsymbol{c}$ 中所在的行
1	无	c_{iikk}	1
2	无	c_{iikk}	k
3	有	c_{ijkl}	$l+k-1$
4	无	c_{ijkl}	$2l+k-2$
N	无	c_{iikk}	i
$2N$	无	c_{iikk}	$2i+k-2$
$3N$	有	c_{ijkl}	$3i+l+k-4$

续表

n_c	对 称 性	c_{ijkl}	在 *c* 中所在的行
$4N$	无	c_{ijkl}	$4i+2l+k-6$
$2N(2N+1)/2$	有	c_{ijkl}	$2j2+j+l+k-4$
	无	c_{ijkl}	$2j2-3j+4i+2l+k-5$
$4N^2$	无	c_{ijkl}	$4N(j-1)+4i+2l+k-6$

【例 26-3】 求解 L 形薄膜的方程 $-\Delta u=1$，$\partial\Omega$ 上为 Dirichlet 边界条件 u=0。最后绘图显示结果。

```
[p, e, t]=initmesh('lshapeg', 'Hmax', 0.2);
[p, e, t]=refinemesh('lshapeg', p, e, t);
u=assempde('lshapeb', p, e, t, 1, 0, 1);
pdesurf(p, t, u)
```

结果如图 26-3 所示。

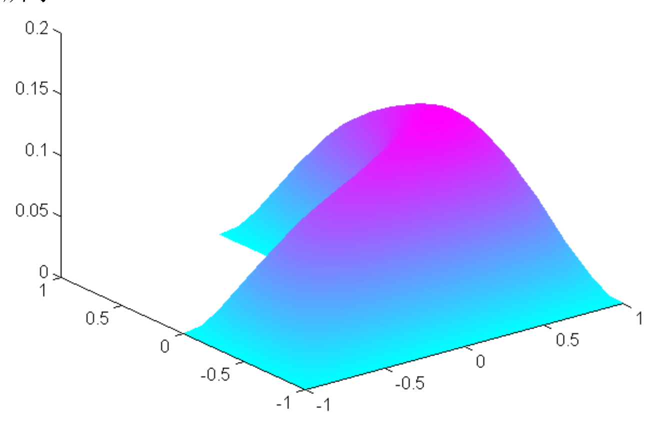


图 26-3 L 形薄膜泊松方程的解

【例 26-4】考虑在原点上有单位点源的单位圆圈的泊松方程。其解析解为：

$$u=-\frac{1}{2\pi}\log(r)$$

定义函数 f=circlef(p,t,u,time)计算右端项。除了原点上的三角形以外，该函数将为所有的三角形返回 0。原点上的三角形将返回值 1/*a*,其中 *a* 为三角形区域。pdedemo 7 函数 M 采用适应性算法演示该问题（见图 26-4）。

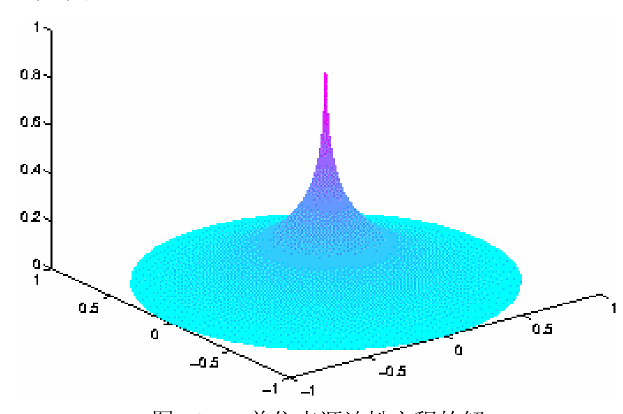


图 26-4 单位点源泊松方程的解

5．hyperbolic 函数

该函数求解双曲线 PDE 问题。其调用格式为：

- u1=hyperbolic(u0, ut0, tlist, b, p, e, t, c, a, f, d) 生成标量 PDE 问题的有限元解

$$\boldsymbol{d}\frac{\partial^2 \boldsymbol{u}}{\partial \boldsymbol{t}^2}-\nabla\cdot(\boldsymbol{c}\nabla\boldsymbol{u})+\boldsymbol{au}=\boldsymbol{f} \qquad \text{在}\Omega\text{上，}$$

或系统 PDE 问题的解

$$\boldsymbol{d}\frac{\partial^2 \boldsymbol{u}}{\partial \boldsymbol{t}^2}-\nabla\cdot(\boldsymbol{c}\otimes\cdot\nabla\cdot\boldsymbol{u})+\boldsymbol{au}=\boldsymbol{f} \qquad \text{在}\Omega\text{上}$$

研究对象为用 p，e 和 t 描述的网格，b 为边界条件，u0 为初值，初始导数为 ut0。

对于标量的情况，解矩阵 u1 中的每一行都是由 p 中对应列给定的坐标上的解。u1 中的每一列为 tlist 中对应项给定的时间上的解。对于具有 np 个节点的 N 维系统，u1 的前 np 列描述 u 的第 1 个元素，接下来的 np 行描述 u 的第 2 个元素，如此类推。这样，u 的元素就作为节点行的 N 个块放到 u 中。

b 描述 PDE 问题的边界条件，它可以是一个边界条件矩阵或边界 M 文件的文件名。边界条件可能与时间 t 有关。边界条件矩阵和边界 M 文件分别描述在 assemb 函数和 pdebound 函数中。

PDE 问题的几何模型由网格数据 p、e 和 t 描述。initmesh 函数对网格数据的意义进行了具体描述。

系数 c，a，d 和 f 可以通过多种方法给定。这些系数也可以与时间 t 有关。在 assempde 函数中可以看到所有选项的列表。

atol 和 rtol 为绝对和相对容限。

- u1=hyperbolic(u0,ut0,tlist,K,F,B,ud,M)生成 PDE 问题的解

$$B'MB\frac{\mathrm{d}^2 u_i}{\mathrm{d}i^2}+K\cdot u_i=F, \qquad u=Bu_i+u_d$$

初值为 u0 和 ut0。

【例 26-5】 求解波动方程：

$$\frac{\partial^2 u}{\partial i^2}=u$$

研究域为方形$-1\leqslant x, y\leqslant 1$ (squareg)，当 $x = \pm 1$ 时，有 Dirichlet 边界条件 $u = 0$；当 $y = \pm 1$（矩形 3）时，有 Neumann 边界条件

$$\frac{\partial u}{\partial n}=0$$

选择

$$u(0) = \arctan(\cos(x))$$

和

$$\frac{\mathrm{d}u(0)}{\mathrm{d}t} = 3\sin(x)\exp(\cos(y))$$

计算时间等于 0, 1/6, 1/3, ... , 29/6, 5 时的解。

```
[p,e,t]=initmesh('squareg');
x=p(1,:)';
```

```
y=p(2,:)';
u0=atan(cos(pi/2*x));
ut0=3*sin(pi*x).*exp(cos(pi*y));
tlist=linspace(0, 5, 31);
uu=hyperbolic(u0, ut0, tlist, 'squareb3', p, e, t, 1, 0, 0, 1);
```

运行后显示：

```
Time: 0.166667
Time: 0.333333
Time: 0.5
Time: 0.666667
Time: 0.833333
Time: 1
Time: 1.16667
Time: 1.33333
Time: 1.5
Time: 1.66667
Time: 1.83333
Time: 2
Time: 2.16667
Time: 2.33333
Time: 2.5
Time: 2.66667
Time: 2.83333
Time: 3
Time: 3.16667
Time: 3.33333
Time: 3.5
Time: 3.66667
Time: 3.83333
Time: 4
Time: 4.16667
Time: 4.33333
Time: 4.5
Time: 4.66667
Time: 4.83333
Time: 5
```

462 次成功计算

70 次计算失败

1066 次函数评价

1 次求偏导数

156 次 LU 分解

1065 个线性系统方程组的解

6．parabolic 函数

利用该函数求解抛物线型 PDE 问题。其调用格式为：

- u1=parabolic(u0,tlist,g,b,p,e,t,c,a,f,d) 生成标量 PDE 问题的有限元解。

$$\boldsymbol{d}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{t}} - \nabla\boldsymbol{\cdot}(\boldsymbol{c}\nabla\boldsymbol{\cdot}\boldsymbol{u}) + \boldsymbol{au} = \boldsymbol{f} \qquad \text{在}\Omega\text{上，}$$

或系统 PDE 问题的解：

$$\boldsymbol{d}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{t}} - \nabla\boldsymbol{\cdot}(\boldsymbol{c}\otimes\boldsymbol{\cdot}\nabla\boldsymbol{\cdot}\boldsymbol{u}) + \boldsymbol{au} = \boldsymbol{f} \qquad \text{在}\Omega\text{上}$$

研究域为由 p，e 和 t 描述的网格,b 为边界条件，初值为 u0。

对于标量的情况，解矩阵 u1 中的每一行都是由 p 中对应列给定的坐标上的解。u1 中的每一列为 tlist 中对应项给定的时间上的解。对于具有 np 个节点的 N 维系统，u1 的前 np 列描述 u 的第一个元素，接下来的 np 行描述 u 的第二个元素，依次类推。这样，u 的元素就作为节点行的 N 个块放到 u 中。

b 描述 PDE 问题的边界条件，它可以是一个边界条件矩阵或边界 M 文件的文件名。边界条件可能与时间 t 有关。边界条件矩阵和边界 M 文件分别描述在 assemb 函数和 pdebound 函数中。

PDE 问题的几何模型由网格数据 p、e 和 t 描述。initmesh 函数对网格数据的意义进行了具体描述。

系数 c，a，d 和 f 可以通过多种方法给定。这些系数也可以与时间 t 有关。在 assempde 函数中可以看到所有选项的列表。

atol 和 rtol 为绝对和相对容限。

- u1=parabolic(u0,tlist,K,F,B,ud,M) 生成 PDE 问题的解

$$B'MB\frac{\mathrm{d}u_i}{\mathrm{d}t} + K\boldsymbol{\cdot}u_i = F, \quad u = Bu_i + u_d$$

u 的初值为 u0。

【例 26-6】求解热传导方程

$$\frac{\partial u}{\partial t} = u$$

研究域为方形 $-1\leqslant x, y\leqslant 1$ (squareg)。在 $x^2 + y^2 < 0.4^2$ 的区域内令 $u(0) = 1$，在其他区域内令 $u(0) = 0$。使用 Dirichlet 边界条件 $u = 0$ (squareb1)。要求计算时间 linspace(0,0.1,20)处的解。

```
[p, e, t]=initmesh('squareg');
[p, e, t]=refinemesh('squareg', p, e, t);
u0=zeros(size(p, 2),1);
ix=find(sqrt(p(1,:).^2+p(2,:).^2)<0.4);
u0(ix)=ones(size(ix));
tlist=linspace(0, 0.1, 20);
u1=parabolic(u0, tlist, 'squareb1', p, e, t, 1, 0, 1, 1);
```

运行后显示：

```
Time: 0.00526316
Time: 0.0105263
Time: 0.0157895
Time: 0.0210526
Time: 0.0263158
Time: 0.0315789
Time: 0.0368421
```

```
Time: 0.0421053
Time: 0.0473684
Time: 0.0526316
Time: 0.0578947
Time: 0.0631579
Time: 0.0684211
Time: 0.0736842
Time: 0.0789474
Time: 0.0842105
Time: 0.0894737
Time: 0.0947368
Time: 0.1
```

96 次成功计算

0 次失败

194 次函数评价

1 次求偏导数

20 次 LU 分解

193 个线性方程组的解

7. pdeeig 函数

利用该函数求解特征值 PDE 问题，其调用格式为：

- [v, l]=pdeeig(b, p, e, t, c, a, d, r) 生成特征值 PDE 问题的有限元解。

$$-\nabla\cdot(c\nabla u)+au=\lambda du \quad 在\Omega上，$$

或系统 PDE 问题的解

$$-\nabla\cdot(c\otimes\nabla u)+au=\lambda du \quad 在\Omega上$$

研究域的几何模型由 p，e 和 t 描述,边界条件由 b 给定。

r 为一二元素向量，指示一个内部实数轴。（右端项可以是-Inf。）算法将给出 l 中该区间内所有的特征值。

v 为一特征值矩阵。对于标量的情况，v 中的每一列都是由 p 中对应节点上解的特征值。v 中的每一列为 tlist 中对应项给定的时间上的解。对于具有 np 个节点的 N 维系统，v 的前 np 列描述 v 的第一个元素，接下来的 np 行描述 v 的第二个元素，如此类推。这样，v 的元素就作为节点行的 N 个块放到 u 中。

b 描述 PDE 问题的边界条件，它可以是一个边界条件矩阵或边界 M 文件的文件名。边界条件可能与时间 t 有关。边界条件矩阵和边界 M 文件分别描述在 assemb 函数和 pdebound 函数中。

PDE 问题的几何模型由网格数据 p，e 和 t 描述。initmesh 函数对网格数据的意义进行了具体描述。

系数 c，a，d 和 f 可以通过多种方法给定。这些系数也可以与时间 t 有关。在 assempde 函数中可以看到所有选项的列表。注意，特征值 PDE 问题是一个均质问题，非均质部分将自动剔除。

PDE 问题的几何模型由网格数据 p，e 和 t 给定。initmesh 函数给出了网格数据意义的细节。

PDE 问题的系数 c，a 和 d 可以通过多种方法给定。assempde 函数给出了一个完整的选项列表。

- [v, l]=pdeeig(K, B, M, r) 生成广义稀疏矩阵特征值问题的解

$$K ui = \lambda B'MBui, u = Bui$$

参数λ的实部位于区间 *r* 内。

【例 26-7】 计算小于 100 的特征值及对应的特征模式。问题为：

$$-\nabla u = \lambda u$$

研究域为 L 形薄膜。然后显示第 1 个和第 16 个特征模式。

```
[p,e,t]=initmesh('lshapeg');
[p,e,t]=refinemesh('lshapeg', p, e, t);
[p, e, t]=refinemesh('lshapeg', p, e, t);
[v, l]=pdeeig('lshapeb', p, e, t, 1, 0, 1, [-Inf 100]);
l(1)                          % 第 1 个特征值
pdesurf(p, t, v(:,1))         % 第 1 个特征模式
figure
membrane(1, 20, 9, 9)        % MATLAB 函数
figure
l(16)                         % 第 16 个特征值
pdesurf(p, t, v(:,16))       % 第 16 个特征模式
```

运行结果为：

```
               Basis= 10,  Time=   9.45,  New conv eig=  2
               Basis= 13,  Time=  11.37,  New conv eig=  2
               Basis= 16,  Time=  13.24,  New conv eig=  3
               Basis= 19,  Time=  15.27,  New conv eig=  4
               Basis= 22,  Time=  17.47,  New conv eig=  5
               Basis= 25,  Time=  19.72,  New conv eig=  6
               Basis= 28,  Time=  22.08,  New conv eig=  7
               Basis= 31,  Time=  24.61,  New conv eig=  7
               Basis= 34,  Time=  27.25,  New conv eig=  9
               Basis= 37,  Time=  29.94,  New conv eig=  9
               Basis= 40,  Time=  32.79,  New conv eig= 11
               Basis= 43,  Time=  35.87,  New conv eig= 11
               Basis= 46,  Time=  39.16,  New conv eig= 16
               Basis= 49,  Time=  42.24,  New conv eig= 17
               Basis= 52,  Time=  45.54,  New conv eig= 20
               Basis= 55,  Time=  49.16,  New conv eig= 28
End of sweep: Basis= 55,  Time=  49.22,  New conv eig= 28
               Basis= 38,  Time=  58.44,  New conv eig=  0
End of sweep: Basis= 38,  Time=  58.50,  New conv eig=  0
ans =
    9.6699
ans =
   93.3166
```

并生成 3 个特征值模式图，如图 26-5、图 26-6 和图 26-7 所示。

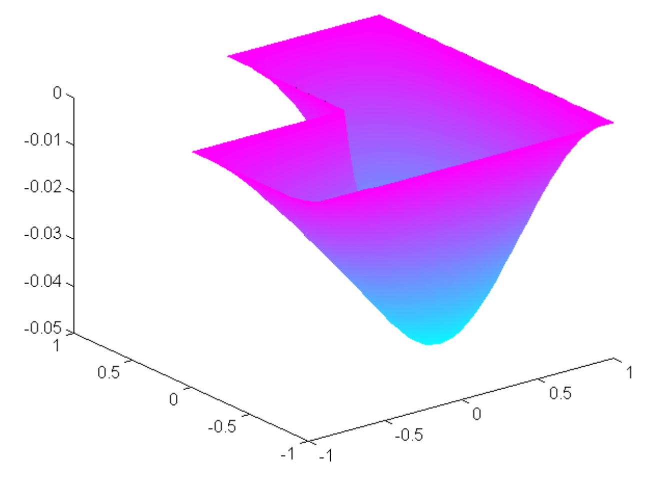


图 26-5　特征值模式图一

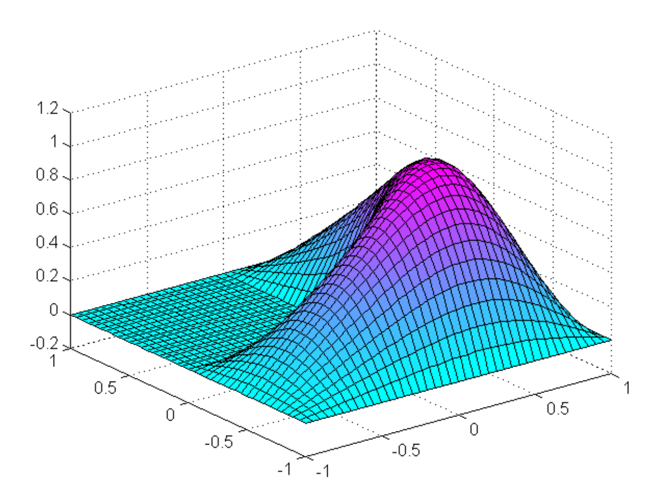


图 26-6　特征值模式图二

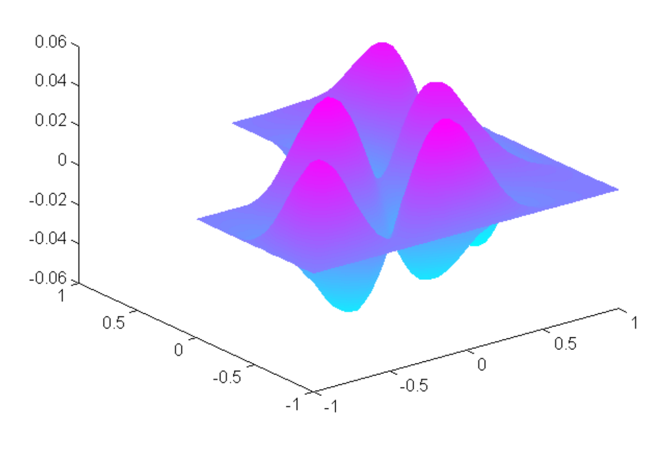


图 26-7　特征值模式图三

对于标准情况，c 和 d 在整个区域上都为正。所有的特征值都为正，将 0 作为下界是一个好的选择。c 和 d 中有一个为 0 的情况作如下讨论：

（1）若某子域中 d = 0，则总体矩阵 M 是奇异的。如果 c 总大于 0，则不会有任何问题。

矩阵束(K,M)将有一系列无限大的特征值。

（2）若子域中 c = 0，则刚度矩阵 K 奇异。矩阵束(K,M)将有很多为 0 的特征值。对于包含 0 的区间，pdeeig 将要花费很长时间才能找到所有为 0 的特征值。建议选择一个比 0 大但比最小非零特征值小的值作为下界。

（3）若有一个区域，其中 c = 0 且 d = 0,将得到一个奇异矩阵束。整个特征值问题是未定的，且任何值看起来都象特征值。

8．pdenonlin 函数

该函数求解非线性 PDE 问题。其调用格式如下所示：

● [u, res]=pdenonlin(b, p, e, t, c, a, f) 求解非线性标量 PDE 问题。

$$-\nabla\cdot(c\nabla u)+au=f \quad 在\Omega上，$$

或非线性系统 PDE 问题：

$$-\nabla\cdot(c\otimes\nabla u)+au=f \quad 在\Omega上$$

其中，系数 c，a 和 f 可能决定于 u。该算法采用阻尼牛顿法进行求解，用 Armijo-Goldstein 法进行一维搜索。

解 u 用解向量 u 代表。assempde 函数中详细说明了解的意义。res 为牛顿逐步残差的范数。

PDE 问题的三角形网格由网格数据 p，e 和 t 给出。

b 描述 PDE 问题的边界条件，它可以是一个边界条件矩阵或边界 M 文件的文件名。边界条件矩阵和边界 M 文件分别描述在 assemb 函数和 pdebound 函数中。注意，一般调用 pdebound 时，边界条件也可以决定于 u。采用定点迭代方案求解非线性边界条件。

系数 c，a 和 f 可以通过多种方法给定。系数可以与时间有关。assempde 函数中给出了格式选项的完整列表。

通过定义表 26-4 所示的选项可以改善求解器。

表 26-4　属性表

属　性　名	属　性　值	默　认　值	描　　述
Jacobian	fixed|lumped|full	fixed	雅可比矩阵的近似值
U0	字符串或数值	0	赋解的初值
Tol	正标量	1e-4	终值处的残差
MaxIter	正整数	25	高斯-牛顿法的最大迭代次数
MinStep	正标量	1/2^16	搜索方向的最小波动
Report	on|off	off	打印收敛信息
Norm	字符串或数值	Inf（+∞）	残差的范数

目前有以下 3 种方法计算雅可比矩阵。

（1）基于 numjac 函数稀疏结构的满秩雅可比矩阵的数值评价。

（2）有限元法数值解。

（3）定点迭代矩阵。其中，通过刚度矩阵求解雅可比矩阵。

通过将 Jacob 属性设置为 full, lumped 或 fixed 来选择需要的方法，更精确的方法则需要更长的计算过程。

U0 为一初值，可以用表达式、标量或向量给出。默认时它设为 0，但对于有 Dirichlet 边

界条件 $u = ex+y$ 的$\nabla(1/u\nabla u) = 0$ 的问题，用 Tol 来控制 Gauss-Newton 迭代法的退出准则，若残差范数小于 Tol，则迭代终止。计算残差的范数通过 Norm 进行选择。

MaxIter 和 MinStep 用于避免迭代过程的死循环。它们分别限制了迭代的最大次数和每次迭代中的最大步长。

若牛顿迭代法不收敛，则给出出错消息：Too many iterations（迭代次数太多）或 Stepsize too small（迭代步长太小）。如果初值为包含 NaN 或 Inf 元素的矩阵，则给出出错消息：Unsuitable initial guess U0 （初值 U0 不合适，默认时，U0=0）。

9．poisolv 函数

利用该函数在矩形网格上对泊松方程进行快速求解，其调用格式如下：

- u=poisolv(b,p,e,t,f) 求解规则矩形网格的带 Dirichlet 边界条件的泊松方程。边界条件 b 中必须为所有边界点指定 Dirichlet 条件。网格 p，e 和 t 必须为规则矩形网格。f 给出泊松方程的右端项。除了 roundoff 错误，应该给出与 u=assempde(b, p, e, t, 1, 0, f)相同的结果。

26.2 自定义界面算法函数

1．pdecirc 函数

利用 pdecirc 函数绘圆，其调用格式为：

- pdecirc(xc,yc,radius) 以(xc,yc)为圆心，以 radius 为半径画圆。如果没有激活 pdetool 集成界面，则它将自动打开，并在空的几何模型中绘制一个圆。
- Pdecirc(xc, yc,radius,label) 选项变量 label 指示了圆的名称（否则选择默认名称）。

PDE Toolbox 中几何描述矩阵的状态被更新为包括该圆。可以通过使用 Draw 菜单中的 Export Geometry Description . . .选项来从 pdetool 中输出几何描述矩阵。几何描述矩阵的格式描述在 decsg 中。

【例 26-8】 下面的命令启动 pdetool 并绘制一个单位圆。

```
pdecirc(0, 0, 1)
```

运行结果如图 26-8 所示。

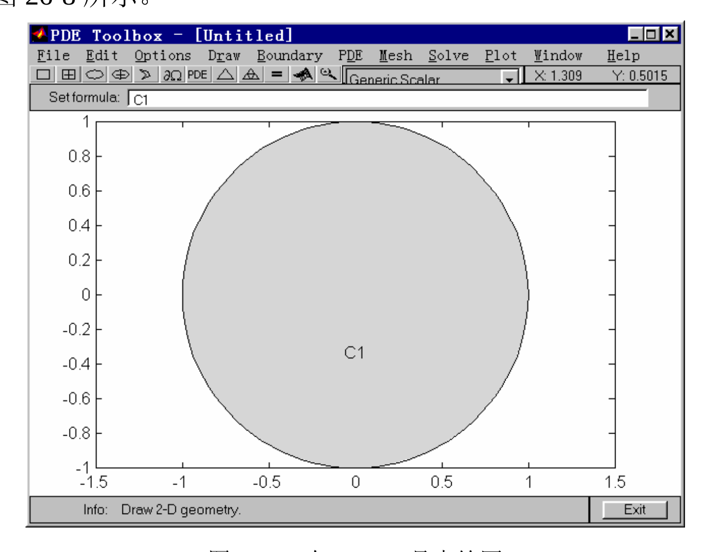


图 26-8 在 PDE 工具中绘圆

2．pdeellip 函数

利用该函数绘制椭圆，其调用格式为：

● pdeellip(xc, yc, a, b, phi, label) 以(xc,yc)为中心，以 a、b 为半轴生成椭圆。给定 phi，旋转椭圆。若 pdetool 集成界面没有激活，则自动启动并在空的几何模型中绘制椭圆。选项变量 label 指定椭圆的名称（否则选择默认的名称）。

PDE Toolbox 中几何描述矩阵的状态被更新为包括该椭圆。可以通过使用 Draw 菜单中的 Export Geometry Description . . .选项来从 pdetool 中输出几何描述矩阵。几何描述矩阵的格式描述在 decsg 中。

【例 26-9】 下面的命令启动 pdetool 并绘制椭圆：

```
pdeellip(0, 0, 1, 0.3, pi/4)
```

3．pdemdlcv 函数

利用该函数将 PDE 工具箱 1.0 模型的 M 文件转换为 PDE 工具箱 1.0.2 版本的格式，其调用格式为：

● pdemdlcv(infile,outfile) 将 PDE 工具箱 1.0 模型的 M 文件转换为 PDE 工具箱 1.0.2 版本的格式。输出文件保存为 outfile。若 outfile 文件名没有.m 扩展名，则系统自动加上。

若试图使用用 PDE 工具箱 1.0 生成的模型 M 文件，必须首先将它们用 pdemdlcv 进行转换。

4．pdepoly 函数

利用该函数绘制多义线，其调用格式为：

● pdepoly(x, y, label) 使用 x，y 定义的坐标绘制多义线。若 pdetool 集成界面没有激活，则它将自动启动，并在空的几何模型中生成多义线。选项变量 label 指定多义线的名称（否则选择默认的名称）。

pdetool 中几何描述矩阵的状态被更新为包括该多义线。可以通过使用 Draw 菜单中的 Export Geometry Description . . .选项来从 pdetool 中输出几何描述矩阵。几何描述矩阵的格式描述在 decsg 中。

【例 26-10】 下面的命令创建一个 L 形（多义线）的薄膜几何模型。

```
pdepoly([-1 0 0 1 1 -1], [0 0 1 1 -1 -1]);
```

5．pderect 函数

利用该函数绘矩形，其调用格式为：

● pderect(xy，label) 绘制矩形，矩形的定义为 xy=[xmin xmax ymin ymax]。

选项变量 label 指定矩形的名称（否则选择默认的名称）。pdetool 中几何描述矩阵的状态被更新为包括该矩形。可以通过使用 Draw 菜单中的 Export Geometry Description . . .选项来从 pdetool 中输出几何描述矩阵。几何描述矩阵的格式描述在 decsg 中。

【例 26-11】

下面的命令序列启动 pdetool 集成界面并通过组合 3 个矩形来得到 L 形薄膜 的几何模型。

```
pderect([-1 0 -1 0])
pderect([0 1 -1 0])
pderect([0 1 0 1])
```

生成的结果如图 26-9 所示。

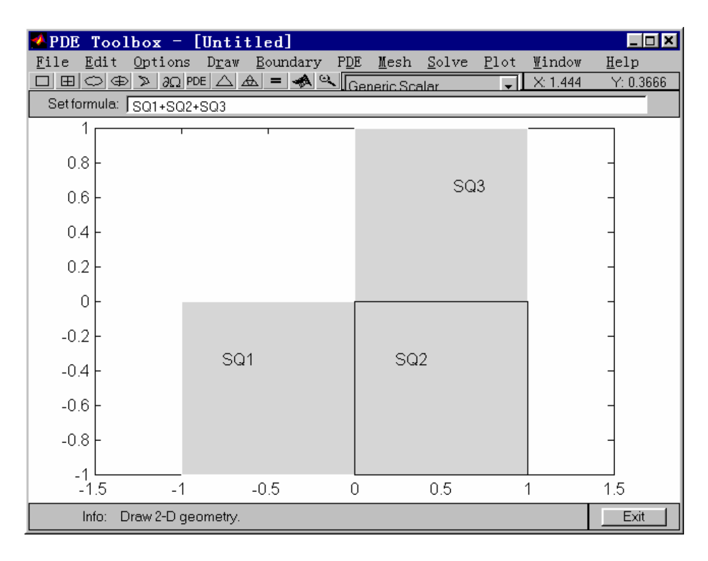


图 26-9　在 PDE 工具中绘矩形

6．pdetool 函数

该函数提供 PDE 工具箱图形用户界面（GUI），其调用格式为：

● pdetool　提供 PDE 工具箱的图形用户界面。调用没有变量的 pdetool 函数将启动该应用程序，用户不能使用带参数的调用方式。

GUI 帮助用户绘制 PDE 问题的二维研究域和边界条件。也可以通过它来指定偏微分方程，创建、检查和细化网格，并计算和显示解。

pdetool 包含一些不同的模式。

在绘图模式中，可以创建建设性实体模型（CSG 模型）。有 4 种实体对象。

- 圆对象：代表圆内的点集；
- 多义线对象：代表多义线内的点集；
- 矩形对象：代表矩形内的点集；
- 椭圆对象：代表椭圆内的点集。

实体对象可以删除和旋转。通过进行多项选择，操作适用于多个成组对象（选择所有选项也可以达到这个目的）。你可以在选择的对象中间进行剪切和粘贴。该模型可以保存。只要键入模型的名称就可以启动 pdetool（它启动了包含创建模型所必要的 MATLAB 命令对应的 M 文件）。

可以通过键入一系列的公式来组合实体对象。每一个对象将被自动赋予惟一的对象名，该对象名显示在实体对象上。默认时，生成的几何模型是所有对象的并集。

GUI 中可以获得"snap-to-grid"函数。它使对象与网格对齐。网格可以显示和取消，比例和网格宽度可以改变。

在边界模式中，可以指定边界条件。在不同的边界上，可能有不同的边界条件。在本模式中，实体对象的原始形状组成模型各子域之间的边界。在本模式中，这些边界可以删除。对象的外边界有不同的颜色，以显示边界类型。红色的外边界对应于 Dirichlet 边界条件，蓝色边界对应 Neumann 边界条件，绿色边界则对应于混合边界条件。可以通过单击 ∂Ω 按钮或通过从 Boundary 菜单中选择 Boundary Mode 选项来返回边界条件。

在 PDE 模式中，可以指定 PDE 问题的类型，以及系数 c, a, f, 和 d。可以单独为每一个子

域指定系数。这样就可以较容易地指定 PDE 模型中的不同材料属性。需要求解的 PDE 问题可以通过单击 PDE 按钮进行指定或通过选择 PDE 菜单中的 PDE Specification . . .选项进行指定。该操作将弹出一个对话框。

在网格模式中，可以控制网格的自动生成和绘制。单击△按钮或在 Mesh 菜单中选择 Initialize Mesh 选项来生成初始网格。单击按钮或在 Mesh 菜单中选择 Refine Mesh 选项可以将网格细化。

在求解模式中，可以指定求解参数并对 PDE 问题求解。对于抛物线 PDE 问题和双曲线 PDE 问题，可以指定初始条件和生成输出的时间。对于特征值问题，可以指定搜索范围。对于椭圆型问题，可以使用适应性求解器和非线性求解器。单击=按钮或在 Solve 菜单中选择 Solve PDE 可以求解。默认时，问题的解将以图形的形式显示在 pdetool 窗口中。

在绘图模式中，可以选择多种可视化方法，如表面图（surface）、网格图（mesh）、等值线图（contour）和矢量图（quiver）等。在所有的绘图类型中，网格都可以隐藏。对于抛物线方程和双曲线方程，当它随时间变化时，可以激活解。可以将解用二维和三维的形式表示。二维图在 pdetool 内部显示。三维图显示在不同的图形窗口中，通过单击对应解的按钮或从 Plot 菜单中选择 Parameters . . .选项来显示不同的图形。该操作打开一个边界条件对话框。

在边界条件对话框中，输入当前选定的边界的边界条件。有下面一些边界条件选项：

- Dirichlet 条件: 边界上 $hu = r$；
- 广义 Neumann 条件：边界上 $\vec{n} \cdot (c\nabla u) + qu = g$；
- 混合条件：Dirichlet 条件和广义 Neumann 条件的混合。

$\vec{n}$ 为向外的单位长度法向向量。

边界条件可以通过多种方法输入。可参见 assemb 函数。

在 PDE 指定对话框中，可以指定 PDE 类型，输入 PDE 系数。有下面类型的 PDE 问题：

- 椭圆型 PDE 问题：

$$-\nabla \cdot (\boldsymbol{c}\nabla \boldsymbol{u}) + \boldsymbol{au} = \boldsymbol{f}$$

- 抛物线型 PDE 问题：

$$\boldsymbol{d}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{t}} - \nabla \cdot (\boldsymbol{cu}) + \boldsymbol{au} = \boldsymbol{f}$$

- 双曲线型 PDE 问题：

$$\boldsymbol{d}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{t}} - \nabla \cdot (\boldsymbol{cu}) + \boldsymbol{au} = \boldsymbol{f}$$

- 特征值 PDE 问题：

$$-\nabla \cdot (\boldsymbol{c}\nabla \boldsymbol{u}) + \boldsymbol{au} = \boldsymbol{du}$$

模型 M 文件：

模型 M 文件包含创建 CSG 模型所必要的 MATLAB 命令。也可以包含其他设置边界条件、定义 PDE 问题、创建网格、求解 PDE 问题和对解绘图的命令。这种类型的 M 文件可以从 File 菜单中保存和打开。

模型 M 文件是 MATLAB 函数而非脚本文件。文件名和模型名必须一致，文件的开始与下面的代码片段近似：

```
function pdemodel
pdeinit;
pde_fig=gcf;
```

```
ax=gca;
pdetool('appl_cb', 1);
setuprop(pde_fig,'currparam', str2mat('1.0', '0.0', '10.0', '1.0'));
pdetool('snapon');
set(ax, 'AspectRatio', [1.5 1]);
set(ax, 'XLim', [-1.5 1.5]);
set(ax, 'YLim', [-1 1]);
set(ax, 'XTickMode', 'auto');
set(ax, 'YTickMode', 'auto');
grid on;
```

pdeinit 命令启动 pdetool。若 pdetool 已经启动，则清除当前模型。接下来的命令创建 pdetool 坐标轴的比例和标记，以及其他用户定义参数。

然后就有一系列绘图命令。可用的命令有 pdecirc, pdeellip, pdepoly 和 pderect。下面的命令序列创建下面的 L 形薄膜，它由 3 个方形实体 SQ1, SQ2, SQ3 组成。

% 几何描述：

```
pderect([-1 0 0 -1], 'SQ1');
pderect([0 1 0 -1], 'SQ2');
pderect([0 1 1 0], 'SQ3');
```

26.3 几何算法函数

1．csgchk 函数

该函数核对几何描述矩阵的有效性。调用格式为：

● gstat=csgchk(gd,xlim,ylim) 核对几何描述矩阵 gd 中的实体对象是否有效，给定实型值 xlim 和 ylim 作为 x 轴和 y 轴的当前长度，并使用多义线的特殊格式。将一条多义线的第一个顶点坐标与最后一个点的顶点坐标重合，可以获得一条闭合的多义线。若 xlim 和 ylim 已经指定了，并且多义线的起点和终点不重合，但如果这些顶点间的距离在一定的“闭合距离”之内，z 则该多义线被认为是闭合的。只有从 pdetool 中调用 csgchk 时才可以使用这些选项输入变量。

● gstat=csgchk(gd)与上面的调用方式等价，除非 decsg 使用的 gd 具有与上面相同的格式。当 csgchk 作为命令行函数时，建议采用这种调用方式。

gstat 是指示实体对象有效状态的整型行向量。

对于圆形实体，gstat=0 说明该圆的半径为正，gstat=1 说明半径非正，gstat=2 说明该圆不是惟一的。

对于多义线，gstat=0 说明多义线是封闭的并且与自身不相交，即它只有惟一的内域；gstat=1 说明该多义线为开放的并且与自身不相交；gstat=2 说明该多义线是闭合且与自身相交的；gstat=3 说明该多义线是开放的，并且与自身相交。

对于矩形实体，gstat 的值对应的情况与多义线的相同。因为矩形可以看成特殊的多义线。

对于椭圆形实体，gstat=0 说明椭圆有正的半轴；gstat=1 说明至少有一个半轴是非正的；gstat=2 说明该椭圆不是惟一的。

若 gstat 完全由 0 组成，则 gd 是有效的，并且可以被 decsg 用做输入变量。

2．csgdel 函数

该函数删除最小子域之间的界线，其调用格式为：

● [dl1, bt1]=csgdel(dl, bt, bl) 删除 bl 表中的边界线段。如果分解几何矩阵的连续性没有通过删除 bl 列表中的元素而得到保留，则子域之间多余的内边界线段将被删除。整个分析域的边界线段不能删除。

内边界线段、边界线段和最小子域的概念在 decsg 函数中解释。

dl 和 dl1 为分解几何矩阵。decsg 函数中有与分解几何矩阵有关的描述。其中还描述了布尔表 bt 和 bt1 的格式。

● [dl1, bt1]=csgdel(dl,bt) 删除所有的边界线段。

3．decsg 函数

该函数将建设性实体几何模型分解为最小子域，其调用格式为：

● decsg 本函数分析用户创建的建设性实体几何模型（CSG 模型）。它分析 CSG 模型，创建一系列不连续的子域，并通过边界线段和内边界线段进行分隔。

图形用户界面 pdetool 使用 decsg 达到多种目的。每一次绘制并改变新的实体对象以后，pdetool 调用 decsg 来绘制实体对象，并将子域正确地最小化。Delaunay 三角化算法 initmesh 也使用 decsg 的输出来生成初始网格。

● dl=decsg(gd, sf, ns) 分解 CSG 模型 gd 到分解几何模型 dl。CSG 模型用几何描述矩阵来代表，分解几何模型由分解几何矩阵代表。decsg 返回最小子域，名称空间矩阵 ns 是一个文本矩阵，它把 gd 中的列和 sf 中的变量名联系起来。

dl=decsg(gd) 返回所有的最小子域。

● [dl, bt]=decsg(gd)和[dl, bt]=decsg(gd, sf, ns) 另外返回一个布尔表，将原始实体对象和最小子域联系起来。bt 中的某列对应于 gd 中的相应列。bt 中的某行对应于一个最小子域指数。

● [dl, bt, dl1, bt1, msb]=decsg(gd)和[dl, bt, dl1, bt1, msb]=decsg(gd, f, ns) 用一个布尔表 bt1 返回第二最小子域。该子域都有相连的边界。这些最小子域可以用 MATLAB 补丁对象图形显示。

（1）几何描述矩阵

几何描述矩阵描述使用 pdetool 画出的 CSG 模型。当前几何描述矩阵可以通过从 Draw 菜单中选择 Export Geometry Description，Set Formula,Labels…选项来获得。

几何描述矩阵中的每一列对应于 CSG 模型中的一个对象。支持 4 种实体对象，对象类型在第一行中指定：

- 对于圆形实体，矩阵第 1 行包含 1，第 2 行和第 3 行分别为 x，y 坐标，第 4 行为圆的半径。
- 对于多义线实体，第 1 行包含 2，第 2 行为多义线边界的线段条数 n。下面的 n 行为边缘起点的 x 坐标。再接下来的 n 行为边缘起点的 y 坐标。
- 对于矩形实体，第 1 行包含 3，其他各行的格式与多义线的格式相同。
- 对于椭圆型实体，第 1 行包含 4，第 2 行和第 3 行分别为 x，y 坐标，第 4 行和第 5 行为椭圆的半轴 a，b。第 6 行包含旋转角度。

sf 包含一系列用 ns 中多个变量表达的公式。运算符号`+'、`*'和 `-' 分别对应于集合运算中的并、交和集合差。其中`+'和`*'的运算优先次序相同，`-'的则更高。可以通过 parentheses 控制运算符号的优先次序。

（2）名称空间矩阵

名称空间矩阵 ns 将 gd 的列和 sf 中的变量名联系起来。ns 中的每一列包含一系列字符，每一个字符列指定一个 gd 中对应几何对象的名称。这样，我们可以再集合公式 sf 指定 gd 中的对象。

（3）分解几何矩阵

分解几何矩阵 dl 包含 decsg 算法创建的不连续最小子域表示的分解几何模型。最小子域的每一个边缘线段对应于 dl 的一列。在每一个这样的列中，第 2 行和第 3 行包含起点和终点的 x 坐标，第 4 行和第 5 行包含起点和终点的 y 坐标。第 6 行和第 7 行包含左右最小子域的标签，标签指示由起点和终点推出的方向（对于圆形和椭圆形线段则按逆时针方向）。最小子域中有 3 种可能的边缘类型：

- 对于圆形边缘线段，第 1 行为 1。第 8 行和第 9 行包含圆心的坐标。第 10 行为圆的半径。
- 对于直线边缘线段，第 1 行为 2。
- 对于椭圆形边缘线段，第 1 行为 4。第 8 行和第 9 行包含椭圆形中心的坐标。第 10 行和第 11 行分别为椭圆的半轴 a 和 b。椭圆的旋转角度保存在第 12 行中。

【例 26-12】 下面的命令序列启动 pdetool 并画一个单位圆和一个边长为一单位的方形。

```
pdecirc(0,0,1)
pderect([0 1 0 1])
```

输入集合公式 C1-SQ1。通过在 Draw 菜单中选择 Export Geometry Description . . .选项将几何描述矩阵，以及名称空间矩阵输出到 MATLAB 主空间中。然后键入：

```
[dl, bt]=decsg(gd, sf, ns);
dl =

2.0000      2.0000      1.0000     1.0000     1.0000
0           0          -1.0000     0.0000     0.0000
1.0000      0           0.0000     1.0000    -1.0000
0           1.0000     -0.0000    -1.0000     1.0000
0           0          -1.0000     0         -0.0000
0           0           1.0000     1.0000     1.0000
1.0000      1.0000      0          0          0
0           0           0          0          0
0           0           0          0          0
0           0           1.0000     1.0000     1.0000
 bt =
     1      0
```

注意，这里有一个最小子域。该子域有 5 个边缘线段、3 个圆形边缘线段和两个直线边缘线段。

本函数采用的算法由下面的步骤组成：

① 确定模型对象内边界之间的交点。

② 对于每一个交点，根据角度和曲度对进入边缘线段进行分类。

③ 确定得到的图形是否是相连的。如果不相连，加上一些相应的边缘，并从步骤 1 开始重新计算。

④ 通过最小子域边缘线段的圆。

⑤ 对于每一个原始区域，确定其中的最小子域。

⑥ 组织输出并剔除额外的边缘。

如果集合公式 sf 不能评价，则返回 NaN。

4．initmesh 函数

该函数创建初始三角形网格，其调用格式为：

● [p, e, t]=initmesh(g) 使用几何指定函数 g 返回三角形网格。它使用 Delaunay 三角化算法。网格的大小由几何模型的形状决定。

g 描述 PDE 问题的几何形态。g 可以是分解几何矩阵或几何 M 文件的文件名。分解几何矩阵和几何 M 文件的格式分别在 decsg 和 pdegeom 中介绍。

输出 p，e 和 t 为网格数据。

在点矩阵 p 中，第 1 行和第 2 行包含网格中各点的 x，y 坐标。

在边缘矩阵 e 中,第 1 行和第 2 行包含起点和终点的参数，第 3 行和第 4 行包含包起点和终点参数，第 5 行包含边缘线段条数，第 6 行和第 7 行为左侧和右侧的子域个数。

在三角形矩阵 t 中，前 3 行包含交点的参数，方向为逆时针，第 4 行包含子域个数。

表 26-5 中的属性名/属性值匹配对是系统允许的：

表 26-5 属性表

属　　性	属　性　值	默　认　值	描　　述
Hmax	numeric	estimate	边缘最大大小
Hgrad	numeric	1.3	网格生长率
Box	on|off	off	保留边界框
Init	on|off	off	边缘三角化
Jiggle	off|mean|min	mean	调用 jigglemesh 函数
JiggleIter	numeric	10	最大迭代次数

Hmax 属性控制网格中三角形的大小。initmesh 创建的三角形的边长不得大于 Hmax。

Hgrad 属性确定 te 网格的生长速率。默认值为 1.3,即生长速率为 30%。Hgrad 必须是介于 1 和 2 之间的数。

Box 属性和 Init 属性与网格算法的计算方式有关。通过设置 Init 属性，可以看到边界的初始三角化。使用下面的命令行：

```
[p, e, t]=initmesh(dl, 'hmax', inf,' init', 'on');
[uxy, tn, a2, a3]=tri2grid(p, t, zeros(size(p, 2)), x, y);
n=t(4, tn);
```

用户可以决定点 xy 处的子域个数 n。如果该点在几何模型之外，则 tn 设置为 NaN，并且命令 n=t(4, tn)失败。

Jiggle 属性用于控制是否对网格进行微调。当三角形的最小个数或平均个数减小时进行网格微调。JiggleIter 可用于设置迭代数的上限。

Initmesh 函数采用 Delaunay 三角化算法，具体步骤是：

① 将节点放到边缘上。

② 在 bounding 对话框中封闭几何图形。

③ 将边缘三角化。

④ 对于边界核对三角化。

⑤ 在大三角形外接圆的圆心中间插入节点。

⑥ 若没有达到 Hmax，则重复步骤④。

⑦ 删除 bounding 对话框。

【例 26-13】 在 pdetool 中创建 L 形薄膜的一个简单三角形网格。

在 pdetool 中做任何事情之前，在 Mesh Parameters 对话框中设置最大边界大小为 inf。打开该对话框的方法是在 Mesh 菜单中选择 Parameters . . .选项。在 Mesh 菜单中还选择项目 Show Node Labels 和 Show Triangle Labels 。然后通过单击△按钮来创建初始三角形（也可以通过在 Mesh 菜单中选择 Initialize Mesh 选项来使网格初始化）。

生成如图 26-10 所示的图形。

通过在 Mesh 菜单中选择 Export Mesh 选项可以将对应的网格数据结构输出到主工作空间中。

```
p
p =
  -1   1   1   0   0  -1
  -1  -1   1   1   0   0
e
e =
   1   2   3   4   5   6
   2   3   4   5   6   1
   0   0   0   0   0   0
   1   1   1   1   1   1
   1   2   3   4   5   6
   1   1   1   1   1   1
   0   0   0   0   0   0
t
t =
   1   2   3   1
   2   3   4   5
   5   5   5   6
   1   1   1   1
```

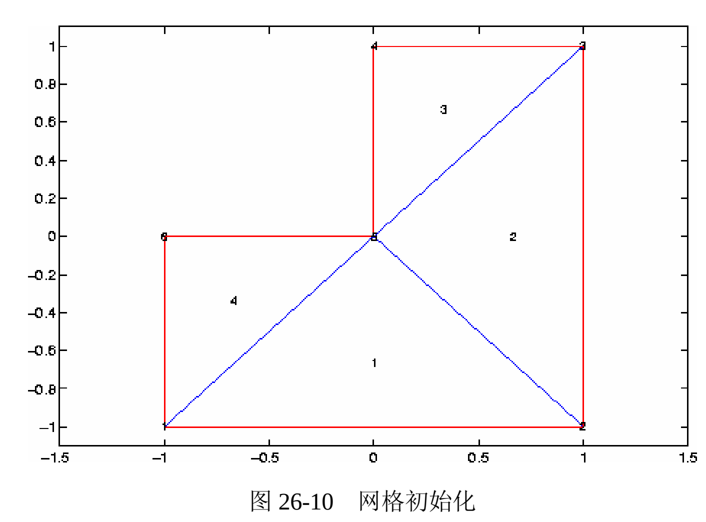


图 26-10　网格初始化

5．jigglemesh 函数

该函数微调三角形网格的内部点，其调用格式为：

● p1=jigglemesh(p,e,t) 通过调节节点的位置来微调网格。网格的数据一般会减少。

表 26-6 中的属性名/属性值匹配对是系统允许的。

表 26-6 属性表

属　　性	属　性　值	默　认　值	描　　述
Opt	off|mean|min	mean	优化方法
Iter	numeric	1 或 20	最大迭代次数

每一个不在边缘线段上的点都向邻近三角形形成的多义线实体中心移动。该过程根据 Opt 变量和 Iter 变量的设置重复操作：

当 Opt 设置为 off 时，本过程重复 Iter 次（默认时 Iter=1）。

当 Opt 设置为 mean 时，本过程重复操作直至平均三角形个数不再增加，或直到达到 Iter 次（默认时为 20）。

【例 26-14】 创建 L 形薄膜的三角形网格。开始没有微调网格，然后微调一次。

```
[p, e, t]=initmesh('lshapeg', 'jiggle',' off');
q=pdetriq(p, t);
pdeplot(p, e, t,' xydata', q,' colorbar', 'on', 'xystyle', 'flat')
p1=jigglemesh(p, e, t, 'opt', 'mean', 'iter', inf);
q=pdetriq(p1, t);
pdeplot(p1, e, t, 'xydata', q, 'colorbar', 'on', 'xystyle', 'flat')
```

生成的图形如图 26-11 所示。

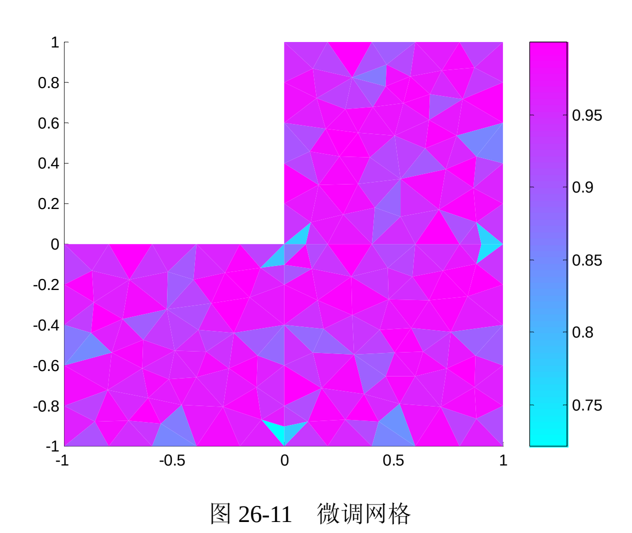


图 26-11 微调网格

6．pdearcl 函数

该函数返回与给定的长度值集合对应的参数曲线的参数值，其调用格式为：

● pp=pdearcl(p, xy, s, s0, s1) 返回与给定的长度值集合对应的参数曲线的参数值。p 为一参数值的单调行向量，xy 为一两行的矩阵，在曲线上给定了对应的点。曲线上的第一个点代表长度值 s0，最后一个点则代表值 s1。返回值 pp 包含 s 中对应于圆弧长度值的参数值。圆弧

长度值 s、s0 和 s1 可以是圆弧长度的 affine 转换。

7. poimesh 函数

利用该函数在矩形几何图形上生成规则网格，其调用格式为：

- [p, e, t]=poimesh(g,nx,ny) 在由 g 指定的矩形几何图形上创建规则网格。通过将"x edge"分为 nx 等份，将"y edge"分为 ny 等份，并在交点上放置(nx+1)*(ny+1)个点。
- [p, e, t]=poimesh(g, n) 使用 nx=ny=n 和[p, e, t]=poimesh(g)及 nx=ny=1。三角形网格通过网格数据 p、e 和 t 进行描述。若 g 不是在描述矩形几何图形，则返回值 p=0。

【例 26-15】 本例返回泊松方程在边界条件由 squareb4 文件给定的矩形网格上的解。可参见 pdedemo8。解的图形见图 26-12。

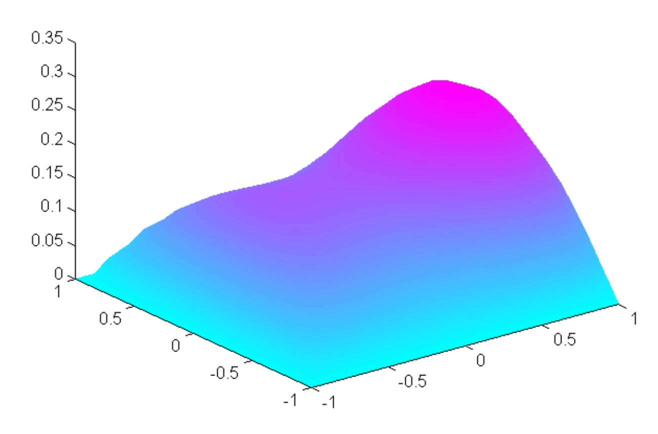


图 26-12 矩形网格上泊松方程的解

8. refinemesh 函数

利用该函数将一个三角形网格细化，其调用格式为：

- [p1, e1, t1]=refinemesh(g, p, e, t) 返回由几何参数 g、点矩阵 p 、边缘矩阵 e 和三角矩阵 t 给定的三角形细化网格。原三角形网格由网格数据 p、e 和 t 决定。详情请参见 initmesh 函数。
- [p1, e1, t1, u1]=refinemesh(g, p, e, t, u) 细化网格并通过线性内插来将函数 u 扩展到新网格。u 的行数应该对应于 p 的列数，p 的列数与 p1 的点数相同。u 的每一列分别内插。

默认的细化方法是均匀细化法。使用该法时，所有指定的三角形分解为四个相同形状的三角形。将最后的参数设置为 longest，可以将细化方法设置为最大边长法。同样，将最后参数设置为 regular 可以将细化方法设置为均匀细化法。有些指定集合之外的三角形也可以细化。

【例 26-16】 细化 L 形薄膜的网格数次，绘出细化后的网格图。

```
[p, e, t]=initmesh('lshapeg', 'hmax', inf);
subplot(2, 2, 1), pdemesh(p, e, t)
[p, e, t]=refinemesh('lshapeg',p, e, t);
subplot(2, 2, 2), pdemesh(p, e, t)
[p, e, t]=refinemesh('lshapeg', p, e, t);
subplot(2, 2, 3),  pdemesh(p, e, t)
[p, e, t]=refinemesh('lshapeg', p, e, t);
subplot(2, 2, 4), pdemesh(p, e, t)
subplot
```

结果如图 26-13 所示。

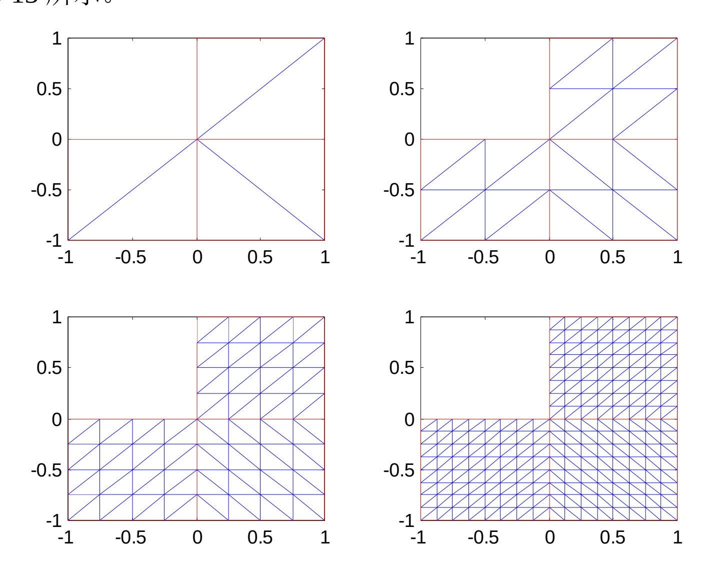


图 26-13 网格的细化

本函数采用的算法分以下步骤：

① 选定将要细化的原始三角形集合。

② 将选定三角形的所有边缘分为两半（均匀细化法）或将最长的边缘分为两半（最长边缘法）。

③ 对有剖分边缘的任何三角形的最长边缘进行剖分。

④ 重复第三步，直到没有更多的边缘可以剖分。

⑤ 引进所有剖分边缘的新点，并通过两个新的入口替换 e 中的所有剖分入口。

⑥ 组成新的三角形。若三角形所有的 3 个边都剖分了，则通过连接各边中点来组成新的三角形。若有两个边被剖分，则长边的中点与对角和另一边的中点相连。如果只有最长的边缘被剖分，则它的中点与对角相连。

9．wbound 函数

利用该函数写边界条件指定文件，其调用格式为：

● fid=wbound(bl, mn) 编写文件名为[mn, '.m']的边界 M 文件。边界 M 文件等价于边界条件矩阵 b1。如果文件不能写，则 fid 参数返回-1。bl 描述 PDE 问题的边界条件。bl 为一边界条件矩阵。详情请参见 assemb 函数。输出文件[mn, '.m']为边界 M 文件的文件名。参见 pdebound 函数。

10．wgeom 函数

利用该函数写几何指定函数，其调用格式为：

● fid=wgeom(dl,mn) 用[mn, '.m']写几何 M 文件的文件名。几何 M 文件等价于分解几何矩阵 dl。如果该文件不能写，则 fid 返回-1。dl 为分解几何矩阵。分解几何矩阵的格式描述在 decsg 中。输出文件[mn, '.m']是几何 M 文件的文件名。几何 M 文件格式在 pdegeom 中进行描述。

26.4 画图算法函数

1．pdecont 函数

利用该函数绘制等值线图，其调用格式为：

- pdecont(p, t, u) 根据 PDE 节点或三角形数据绘制 u 的等值图。
- h=pdecont(p, t, u) 同上，还返回轴对象的句柄。

若 u 为一列向量，则用节点数据。若 u 是行向量，则用三角形数据。使用函数 pdeprtni，可以将三角形数据转换为节点数据。

PDE 问题的几何形态通过网格数据 p 和 t 给定。在 initmesh 函数中可以查看网格数据意义的细节。

- pdecont(p, t, u, n) 使用 n 水平绘图。
- pdecont(p, t, u, v) 使用 v 指定的水平绘图。本命令是调用下面函数的快捷键：
- pdeplot(p, [ ], t, 'xydata', u, 'xystyle', 'off', 'contour', 'on', 'levels', n, 'colorbar', 'off'); 如果希望对等值线图有更多的控制，则使用 pdeplot 代替 pdecont。

【例 26-17】　绘制 L 形薄膜上方程$-\Delta u=1$解的等值线图。在$\partial\Omega$上使用 Dirichlet 边界条件。

```
[p, e, t]=initmesh('lshapeg');
[p, e, t]=refinemesh('lshapeg', p, e, t);
u=assempde('lshapeb', p, e, t, 1, 0, 1);
pdecont(p, t, u)
```

生成的图形如图 26-14 所示。

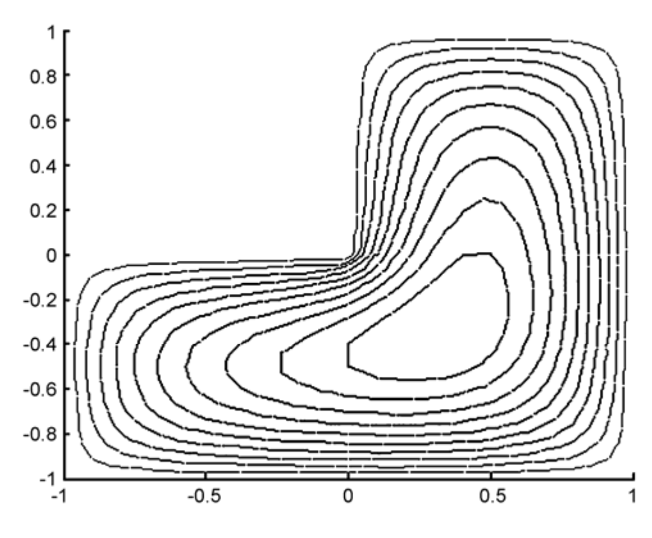


图 26-14　L 形薄膜上方程解的等值线图

2．pdegplot 函数

利用该函数绘制 PDE 几何图，其调用格式为：

- pdegplot(g) 绘制 PDE 问题的几何图形。

● h=pdegplot(g) 返回轴对象的句柄。

g 描述 PDE 问题的几何形态。g 可以是一个分解几何矩阵或几何 M 文件的文件名。分解几何矩阵和几何 M 文件的格式分别描述在 decsg 和 pdegeom 中。

【例 26-17】 绘制 L 形薄膜的几何图，如图 26-15 所示。

```
pdegplot('lshapeg')
```

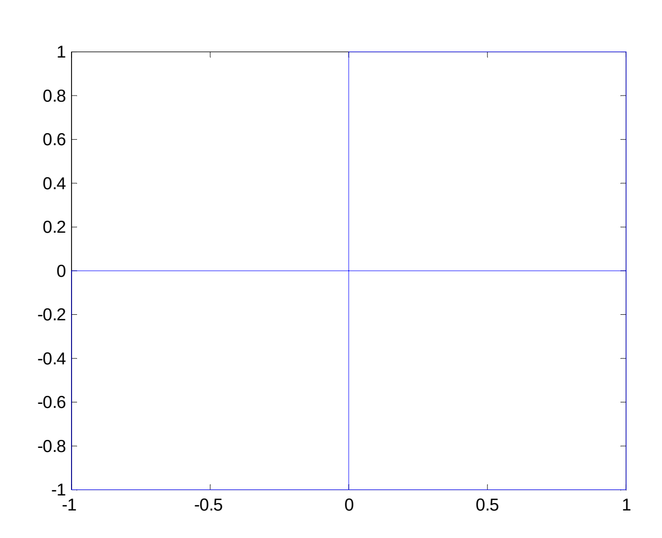


图 26-15　L 形薄膜的几何图形

3．pdemesh 函数

利用该函数绘制 PDE 三角形网格，其调用格式为：

● pdemesh(p, e, t) 绘制由网格数据 p，e 和 t 指定的网格。

● h=pdemesh(p, e, t) 还返回图的轴对象的句柄。

● pdemesh(p, e, t, u) 使用 PDE 节点或三角形数据绘制 u 的网络图。若 u 为一列向量，则用节点数据。如果 u 是行向量，则用三角形数据。该命令绘图比用 pdesurf 命令要快。

PDE 问题的几何形状通过网格数据 p，e 和 t 给定。在 initmesh 函数中可以看到有关网格数据意义的细节。

该命令是下面调用的快捷键：

```
pdeplot(p, e, t)
pdeplot(p, e, t, 'zdata', u)
```

如果希望更好地控制网格绘图，使用 pdeplot 代替 pdemesh。

【例 26-19】 绘制 L 形薄膜几何形态网格，如图 26-16 所示。

```
[p, e, t]=initmesh('lshapeg');
[p, e, t]=refinemesh('lshapeg', p, e, t);
pdemesh(p, e, t)
```

现在在 L 形薄膜的几何模型上求解泊松方程 $-\Delta u = 1$。在 $\partial\Omega$ 上使用 Dirichlet 边界条件 $u = 0$，并绘制结果图，如图 26-17 所示。

```
u=assempde('lshapeb', p, e, t, 1, 0, 1);
pdemesh(p, e, t, u)
```

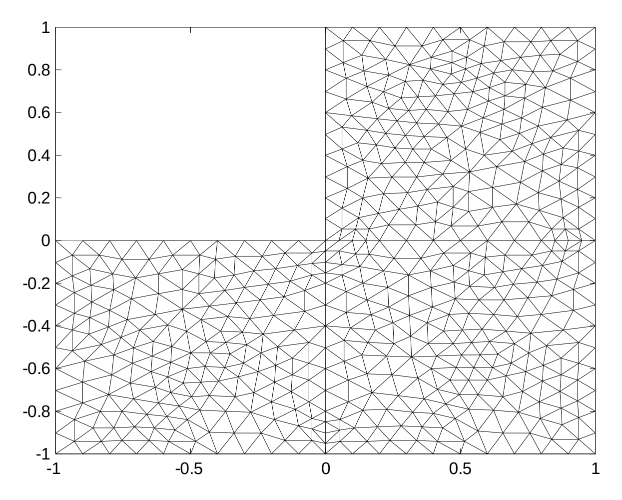


图 26-16　L 形薄膜的三角形网格图

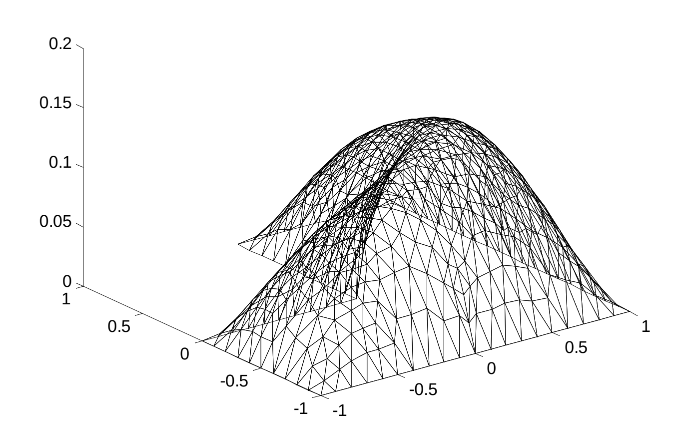


图 26-17　L 形薄膜上泊松方程的解

4．pdeplot 函数

一般 PDE 工具箱绘图函数，其调用格式为：

● pdeplot(p, e, t, p1, v1,...) 为一般 PDE 工具箱绘图函数。它可以同时显示几个函数的 PDE 解。PDE 问题的几何形式通过网格数据 p，e 和 t 决定。细节内容请参见 initmesh 函数。

有效的属性/属性值匹配对如表 26-7 所示。

表 26-7　属性表

属　性　名	属性值/默认值	描　　述
xydata	data	三角形数据
xystyle	off|flat|{interp}	x-y 数据绘图类型
contour	{off}|on	显示等值线图
zdata	data	节点或三角形数据
zstyle	off|{continuous}| discontinuous	三维图类型
flowdata	data	节点或三角形数据

续表

属　性　名	属性值/默认值	描　　述
flowstyle	off|{arrow}	流动图类型
colormap	colormap cool	x-y 数据彩色图的名称或矩阵
xygrid	{off}|on	在出图之前转换为 x-y 网格
gridparam	[tn; a2; a3]	调用 tri2grid 函数时的三角形指数和内插参数
mesh	{off}|on	在图中显示网格
colorbar	off|{on}	显示色彩比例条
title	off|{on}	输出标题文本
levels	10	水平个数或指定水平的向量

pdeplot 函数可以在 pdetool GUI 内部和通过命令行使用。它可以同时显示 3 个实体。xydata 可以用表面图可视化显示。平面或内插阴影都可以用于表面图。在表面图上可以叠加等值线图（用黑色表示）或通过设置 contour 为 on 来独立绘出等值线图(用颜色表示)。zdata 可以通过高度显示来可视化。数据流可以用箭头图可视化表示。所有的数据类型可以是节点数据或三角形数据（数据流只能是三角形数据）。节点数据用长度 size(p,2)的列向量代表的节点数据，三角形数据用长度 size(t,2)数据的行向量代表。若没有 xydata、zdata 或者 flowdata 已经提供了，则 pdeplot 根据 p，e 和 t 绘网格图。

mesh 选项控制是显示三角形网格还是隐藏（默认选项）它。选项 xygrid 首先将数据转换为 x–y 数据(使用 tri2grid 函数),然后使用标准的 MATLAB 绘图算法。gridparam 参数将 tri2grid 数据传到 pdeplot 函数。Colorbar 参数控制图中色彩比例条的添加。输入 title 参数则可以给当前图形输入标题。levels 参数仅适用于等值线图。

- h=pdeplot(p, t, u) 另外返回坐标系中对象的句柄。

【例 26-20】 下面的命令序列将 L 形薄膜的泊松方程的解绘成三维图，如图 26-18 所示。

```
[p, e, t]=initmesh('lshapeg');
u=assempde('lshapeb', p, e, t, 1, 0, 1);
pdeplot(p, e, t, 'xydata', u, 'zdata', u, 'mesh', 'off');
```

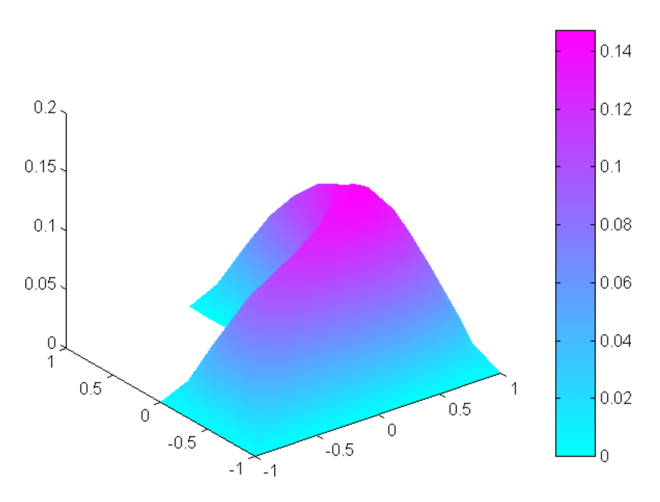


图 26-18　L 形薄膜上方程解的三维图

5．pdesurf 函数

利用该函数绘三维表面图，其调用格式为：

● pdesurf(p, t, u)　绘制 PDE 节点或三角形数据的三维表面图。若 u 为列向量，则用节点数据且使用了连续类型和内插阴影；若 u 是行向量，则用三角形数据，且使用了连续类型和平面阴影。

● h=pdesurf(p, t, u)　另外返回轴对象的句柄。对于节点数据，该命令是调用下面函数的快捷键。

```
pdeplot(p, [ ], t, 'xydata', u, 'xystyle', 'interp',...
        'zdata', u,'zstyle', 'continuous', 'colorbar', 'off');
```

并且对于三角形数据，有

```
pdeplot(p, [ ], t, 'xydata', u, 'xystyle', 'flat',...
        'zdata', u, 'zstyle', 'discontinuous', 'colorbar', 'off');
```

若需要对表面图有更多的控制，则使用 pdeplot 函数替换 pdesurf 函数。

【例 26-21】　生成 L 形薄膜的几何图形上方程$-\Delta u = 1$的解的表面图。在$\partial\Omega$上使用 Dirichlet 边界条件。

```
[p, e, t]=initmesh('lshapeg');
[p, e, t]=refinemesh('lshapeg', p, e, t);
u=assempde('lshapeb', p, e, t, 1, 0, 1);
pdesurf(p, t, u)
```

26.5　实用算法函数

1．Dst，idst 函数

利用它们进行离散化 sin 转换，其调用格式为：

● y=dst(x) 计算 x 的列的离散 sin 转换。将 x 中的行数设置为 2m − 1 可以获得最佳的计算速度。

● y=dst(x, n) 在转换之前将向量 x 斜截到长度为 n。

若 x 为矩阵，dst 操作适用于每一列。

● x=idst(y) 计算 y 的列的逆 sin 转换。将 y 的行数设置为 2m-1 时将获得最佳的计算速度。

● x=idst(y, n) 在转换之前将向量 y 斜截到长度为 n。

若 y 为矩阵，idst 操作适用于每一列。

2．pdeadgsc 函数

利用 pdeadgsc 函数，使用相对容限临界值选择三角形，其调用格式为：

● bt=pdeadgsc(p, t, c, a, f, u, errf, tol) 在 bt 中返回需要细化的三角形指数。PDE 问题的几何形态由网格数据 p 和 t 给定。参见 initmesh 函数。c、a 和 f 为 PDE 系数。细节参见 assempde 函数。u 为当前解，用列向量给出。细节参见 assempde 函数。errf 为误差指示器，由 pdejmps 函数决定。tol 为容限参数。三角形用临界值 errf>tol*scale 进行选择,其中 scale 由下式计算：

scale=max(fmax*l^2, amax*umax*l^2, cmax*umax)

其中，cmax, amax, fmax 和 umax 分别为 c, a, f 和 u 的最大值，l 为包含几何图形的最小方形的边。

然后。使 tol 参数独立于方程和几何图形的缩放。

3．pdeadworst 函数

利用该函数选择相对于最坏值的三角形，其调用格式为：

● bt=pdeadworst(p, t, c, a, f, u, errf, wlevel) 返回将要细化的三角形的指数到 bt 中去。PDE 问题的几何图形由网格数据 p 和 t 给定。细节参见 initmesh 函数。c, a 和 f 为 PDE 系数。细节参见 assempde 函数。u 为当前解，用列向量给出。细节参见 assempde 函数。errf 参数为误差指示器，由 pdejmps 函数计算。wlevel 为与最坏误差相关的误差水平。wlevel 必须界于 0 和 1 之间。

三角形用临界条件 errf>wlevel*max(errf)给定。

4．pdecgrad 函数

该函数计算 PDE 解的变化，其调用格式为：

● [cgxu, cgyu]=pdecgrad (p, t, c, u) 返回每一个三角形中心处的 flux 和 c⊗∇u。

$$\sum_{j=1}^{N} c_{ij11}\frac{\partial u_j}{\partial x}+c_{ij12}\frac{\partial u_j}{\partial y}$$

cgxu 中的第 i 行包含：

$$\sum_{j=1}^{N} c_{ij12}\frac{\partial u_j}{\partial x}+c_{ij22}\frac{\partial u_j}{\partial y}$$

cgyu 中的第 i 行包含：

在 cgxu 和 cgyu 中，t 内的每一个三角形都对应一列。

PDE 问题的几何图形由网格数据 p 和 t 给定。细节请参见 initmesh 函数。

PDE 问题中的 c 系数可以通过多种方法给定。在 assempde 中有一个所有选项的完整列表。

解向量 u 的格式在 assempde 函数中描述。

若 c 决定于 t，则标量选项参数 time 用于抛物线型问题和双曲线型问题。

5．pdeent 函数

该函数给定三角形集合相邻的三角形的指数，其调用格式为：

● ntl=pdeent(t, tl) 给定三角形数据 t 和三角形指数列表 tl，计算三角形的指数 nt1。

6．pdegrad 函数

该函数计算 PDE 解的梯度，其调用格式为：

● [ux, uy]=pdegrad(p, t, u) 返回在每一个三角形的中心进行评价的 u 的梯度。

$$\frac{\partial u_i}{\partial x}$$

ux 的第 i 行中从 1 到 N 包含：

$$\frac{\partial u_i}{\partial y}$$

uy 的第 i 行从 1 到 N 包含。

在 ux 和 uy 中，对于 t 内的每一个三角形都有一列。PDE 问题的几何图形由网格数据 p 和 t 给定。细节请参见 initmesh 函数。解向量 u 的格式在 assempde 函数中描述。选项变量 sdl 将计算限于 sdl 列表中的子域。

7．pdeintrp 函数

利用该函数，从节点数据至三角形中点数据进行内插，其调用格式为：

● ut=pdeintrp(p, t, un) 给出三角形节点至中点的线性插值。PDE 问题的几何图形由网格数据 p 和 t 给定。细节请参见 initmesh 函数。令 N 为 PDE 系统的维数，np 为节点个数，nt 为三角形个数。节点数据中的元素作为长度 np 中的 N 列或一般解向量保存在 un 中。un 中的第 1 个 np 值描述第 1 个元素，第 2 个 np 值描述第 2 个元素，如此类推。三角形数据的元素保存在 nt 的 N 行中。

需要注意的是，pdeprtni 函数和 pdeintrp 函数不是逆函数。内插将导致均化。

8．pdejmps 函数

该函数对于自适应网格进行误差估计，其调用格式为：

● errf=pdejmps(p, t, c, a, f, u, alfa, beta, m) 计算用于 adaption 的误差指示函数。errf 的列对应于三角形，各行则对应于 PDE 系统的不同方程。p 和 t 为网格数据。细节参见 initmesh 函数。c、a 和 f 为 PDE 系数。细节参见 assempde 函数。c、a 和 f 必须扩展，以使列对应于三角形。u 为解向量。细节参见 assempde 函数。计算每一个三角形 K 的误差指示器 E(K)的公式为：

$$E(K)=\alpha\|h(f-\alpha u)\|_K+\beta\left(\frac{1}{2}\sum_{r\in\partial K}h_r^2[n_r\cdot(c\nabla u_h)]^2\right)^{1/2}$$

9．pdeprtni 函数

该函数从三角形中点数据向节点数据进行内插，其调用格式为：

●un=pdeprtni(p,t,ut) 给出从三角形中点至节点的线性插值。PDE 问题的几何形态由网格数据 p 和 t 给出。更多细节请参见 initmesh 函数。令 N 为 PDE 系统的维数，np 为节点个数，nt 为三角形的个数。ut 中三角形数据的元素保存在长度 nt 的 N 行中。节点数据的元素作为长度 np 的 N 列保存在 un 中。

10．Pdesdp, pdesde 和 pdesdt 函数

这 3 个函数分别计算子域集合中点/边缘/三角形的指数，其调用格式为：

● [i, c]=pdesdp(p, e, t, sdl) 给定网格数据 p，e 和 t 以及子域个数列表 sdl，该函数返回属于那些子域的所有点。一个点可以属于几个子域，属于 sdl 子域中的点被分为两个不连续的集合。i 包含 sdl 列表中所有子域的点的指数。c 则列出了也属于其他子域的点的列表。

● c=pdesdp(p, e, t, sdl) 返回 sdl 中属于一个以上点的指数。

● i=pdesdt(t, sdl) 给定三角形数据 t 和子域个数 sdl 的列表，i 包含子域集合中三角形的指数。

● i=pdesde(e, sdl) 给定边缘数据 e，它分离子域集合外边界边缘的指数。

若没有给定 sdl，则用所有子域的列表。

11．pdesmech 函数

该函数计算结构力学张量函数，其调用格式为：

● ux=pdesmech(p, t, c, u, p1, v1,...) 返回每个三角形中心处的张量表达。张量表达为平面应力和平面应变条件下的结构力学应力应变。在输出解、网格和 PDE 系数到 MATLAB 主空间以后，pdesmech 函数对利用 pdetool GUI 结构力学应用模式进行计算得到的解进行后处理。

必须提供泊松比 nu,以计算剪应力和剪应变，以及平面应变模式中的 von Mises 有效应力。

有效的属性名/属性值匹配对如表 26-8 所示。

表 26-8　属性表

属　性　名	属性值/默认值	描　　述
tensor	ux|uy|vx|vy|exx|eyy|exy|sxx|syy|sxy|e1|e2|s1|s2|{von Mises}	张量表达
application	{ps}|pn	平面应力/平面应变
Nu	标量或字符串表达{0.3}	泊松比

可用的张量表达包括：

$ux = \dfrac{\partial u}{\partial x}$

$uy = \dfrac{\partial u}{\partial y}$

$vx = \dfrac{\partial u}{\partial x}$

$vy = \dfrac{\partial u}{\partial y}$

exx, x 方向的应变（ε_x）

eyy, y 方向的应变（ε_y）

exy, 剪应变（γ_{xy}）

sxx, x 方向的应力（σ_x）

syy, y 方向的应力（σ_y）

sxy, 剪应力（τ_{xy}）

e1, 第 1 主应变（ε1）

e2, 第 2 主应变（ε2）

s1, 第 1 主应力（σ1）

s2, 第 2 主应力（σ2）

von Mises, 冯 • 米色斯有效应力

【例 26-22】　假设用应用模式"Structural Mechanics, Plane Stress,"求解问题，解为 u，网格数据为 p 和 t，PDE 系数 c 都被输出到 MATLAB 主空间，则 x 方向的应变做如下计算：

```
sx=pdesmech(p, t, c, u, 'tensor', 'sxx');
```

令泊松比等于 0.3，计算平面应变问题的 von Mises 有效应力，键入

```
mises=pdesmech(p, t, c, u, 'tensor', 'von Mises',...
'application', 'pn', 'nu', 0.3);
```

12．pdetrg 函数

该函数返回三角形的几何数据，其调用格式为：

- [ar, a1, a2, a3]=pdetrg(p, t) 返回每一个三角形的面积到 ar 中，返回每一个三角形负余切的一半到 a1, a2, a3 中。
- [ar, g1x, g1y, g2x, g2y, g3x, g3y]=pdetrg(p, t) 返回三角基函数的面积和梯度元素。

● PDE 问题的三角形网格由网格数据 p 和 t 给定。在 initmesh 函数中有网格数据意义的详细内容。

13．pdetriq 函数

该函数返回三角形的质量状况，其调用格式为：

● q=pdetriq(p, t) 返回由网格数据给定的三角形的质量状况。

三角形网格由网格数据 p，e 和 t 给定。细节内容参见 initmesh 函数。

14．poiasma 函数

用于泊松方程快速求解器的边界点矩阵，其调用格式为：

● K=poiasma(n1, n2, h1, h2) 组合边界点对刚度矩阵的贡献。n1 和 n2 为第 1 和第 2 方向上的点数。h1 和 h2 为网格的间隔。K 为稀疏 n1*n2-by-n1*n2 矩阵。

● K=poiasma(n1, n2) 此时 h1=h2。

● K=poiasma(n) 此时 n1=n2=n。

15．poicalc 函数

该函数为矩形网格上泊松方程的快速求解器，其调用格式为：

● u=poicalc(f,h1,h2,n1,n2) 计算均匀间隔的网格内点的泊松方程的解。u 的列包含对应于 f 的右端项的列的解。h1 和 h2 为第 1 方向和第 2 方向上的间隔，n1 和 n2 为点数。f 中的行数必须为 n1*n2。若 n1 和 n2 没有给定，则假定为 f 的行数的平方根。若 h1 和 h2 没有给定，则假设它们相等。通过在第 1 个方向上进行 sin 转换以及第 2 个方向上的三对角矩阵可以得到问题的解。当 nl 为 1 时，计算效果最佳。

16．poiindex 函数

该函数计算经过规范排序的矩形网格的点的索引号，其调用格式为：

● [n1, n2, h1, h2, i, c, ii, cc]=poiindex(p, e, t, sd) 确定给定的子域 sd 中的网格 p、e、t 为均匀间隔的矩形网格。若网格不是规则网格，则 n1 的返回值为 0。否则 n1 和 n2 为第 1 方向和第 2 方向上的点号，h1 和 h2 为间隔。i 和 ii 为长度(n1–2)*(n2–2)并包含内点的索引号。i 包含原始网格的索引号，而 ii 则包含规范顺序的索引号。c 和 cc 长度为 n1*n2–(n1–2)*(n2–2)并包含边界点的索引号。ii 和 cc 是持续增长的。

在标准排序中，点号先按从左到右的顺序排列，然后按从底到顶排列。这样，若 n1=3、n2=5,则 ii=[5 8 11]且

cc=[1 2 3 4 6 7 9 10 12 13 14 15].

17．sptarn 函数

该函数求解广义稀疏特征值问题，其调用格式为：

● [xv, lmb, iresult] = sptarn(A, B, lb, ub, spd, tolconv, jmax, maxmul) 找到区间[lb，ub]上 $(A - \lambda B)x = 0$ 的特征值。(线性多项式矩阵 $\boldsymbol{A}_{ij} - \lambda\boldsymbol{B}_{ij}$, $\boldsymbol{A} - \lambda\boldsymbol{B}$ 称为束。)

A 和 B 为稀疏矩阵。lb 和 ub 为进行求解的特征值的上界和下界。如果 ub 左侧的所有的特征值都找到了，则我们可以认为有 lb=-inf。lb 和 ub 中必须有一个是有限的。如果求解区间更窄，则求解速度更快。在复数情况下，lmb 的实数部分可以比作 lb 和 ub。

xv 为特征向量，经过了排序，这样范数(a*xv-b*xv*diag(lmb))很小。lmb 是经过排序的特征值。若 iresult>=0，则算法继续，并且区间内所有的特征值都可以找到。若 iresult<0，则算

法不成功，可能有更多的特征值—缩小区间再试一试。

若已知束是对称正定的，则 spd 为 1（默认时为 0）。

tolconv 为期望的相对精度。默认值为 100*eps,其中 eps 为机器精度。

jmax 为基向量的最大个数。该算法需要 jmax*n 大小的工作空间，这样，在小型机上需要做一些小的调整，否则令 jmax=100 为默认值。正常情况下，当特征值收敛时，算法早已终止。

maxmul 为 Arnoldi 试运行次数。它必须至少与任何特征值的最大乘子一样大。若给定了 jmax 的一个小值，则需要许多 Arnoldi 运行。其默认值为 maxmul=n, 当要求单位矩阵的所有特征值时需要用到它。

Arnoldi 算法使用了谱转换。该转换在 ub 或 lb 上进行，当边界无限时，可以是（lb,ub）上的任意一点。Arnoldi 运行的步数 j 决定于区间中有多少特征值，但它在 j=min(jmax,n)处终止。终止以后， 该算法重新开始计算，并在所有已经找到的正交补集中求得更多的 Schur 向量。当在区间 lb < lmb <= ub 中没有更多的特征值时，该算法终止。对于 jmax 的小值，要得到某个特定的特征值可能还需要重算几次。当 jmax 至少比区间内的特征值个数大 1，该算法开始计算，但后面还需要多次重算。对于 jmax 的大值，需要运行 mu1+1 次。它是一个很好的选择。mul 为区间内某特征值的最大乘子。

需要注意的是，该算法在不对称束和对称束上都可以运行。进行因子分解时，参数 spd 只需在 symmmd 和 colmmd 之间进行选择。

如果出现下面的问题，给出相应的对策：

① 若收敛很慢，试给一个更小的区间 lb 和 ub，给一个更大的 jmax，更大的 maxmul，并重新计算。

② 若因子分解失败，则将 lb 或 ub 设置为有界，再重新计算。然后，在随机点上进行选择转换，而不是在如我们所期望的特征值点上。若还失败，则核对束是否奇异。

③ 若计算一直进行下去，则可能有太多的特征值。试给一个小值 maxmul=2，看得到哪个特征值。这样你可以得到一些特征值，但一个负的 iresult 值显示得到的不是所有的特征值。

④ 若内存溢出，则给一个小的 jmax。

⑤ 该算法是针对接近实数轴的特征值的，若希望靠近虚数轴，则令 A=i*A。

⑥ 当 spd=1 时，转换在 lb 处进行，这样，对于对称正定矩阵，将获得更快的因子分解速度。这样设置不会导致大的错误，但是如果 lb 比最小的特征值大，则运行速度要减慢。

18．tri2grid 函数

该函数从 PDE 三角形网格到矩形网格进行内插，其调用格式为：

● uxy=tri2grid(p, t, u, x, y) 根据由 p，t 定义的三角形网格上的函数值 u，计算由向量 x 和向量 y 定义的网格上的函数值 uxy。这些值是在包含网格点的三角形上进行线性内插得到的。向量 x 和 y 必须是递增的。

● [uxy, tn, a2, a3]=tri2grid(p, t, u, x, y) 另外列出包含每一个网格点的三角形的指数 tn，以及内插系数 a2 和 a3。

● uxy=tri2grid(p, t, u, tn, a2, a3)，其中 tn，a2 和 a3 在 tri2grid 的早期调用中进行计算，并使用同样的网格进行内插。对于三角形网格以外的网格点，参数 uxy，tn，a2 和 a3 返回 NaN。

26.6 自定义算法函数

1．pdebound 函数

该函数定义边界条件 M 文件，其调用格式为：

● [q, g, h, r]=pdebound(p, e, u, time)　边界 M 文件指定 PDE 问题的边界条件。最一般的边界条件形式可以表达如下：

$$hu = r$$

$$\vec{n} \cdot (c \otimes \nabla u) + \underline{q} u = g + h'\mu$$

$\vec{n} \cdot (c \otimes \nabla u)$为 N*1 的矩阵，其（i，1）元素为

$$\sum_{j=1}^{N}\left(\cos(a)c_{i,j,1,1}\frac{\partial}{\partial x}+\cos(a)c_{i,j,1,2}\frac{\partial}{\partial y}+\sin(a)c_{i,j,2,1}\frac{\partial}{\partial x}+\sin(a)c_{i,j,2,2}\frac{\partial}{\partial y}\right)u_j$$

式中，$\vec{n}=(\cos(a),\sin(a))$为边界上向外的方向向量。有 M 个 Dirichlet 条件，h 是 M*N 矩阵，其中 $M \geqslant 0$。

对于 $M = 0$，我们认为是广义 Neumann 边界条件,对于 $M = N$，为一 Dirichlet 条件,当 $0 < M < N$ 时，为混合边界条件。

边界 M 文件[q,g,h,r]=pdebound(p, e, u, time)在边缘几何 e 上计算 q、g、h 和 r 的值。

矩阵 p 和 e 为网格数据。e 只需要为网格中边缘的子集。

输入变量 u 和 time 分别用于非线性求解器和时步算法。如果 u 和 time 对应的参数没有传给 assemb，则它们为空矩阵。如果 time 为 NaN 且任何函数 q，g，h 和 r 取决于 time，则 pdebound 函数必须返回大小正确的矩阵，且在对应的输出变量中所有的位置上包含 NaN。

解用向量 u 表示。q 和 g 必须包含每一条边界中点上的 q 值和 g 值。这样，我们有 size(q)=[N^2 ne]，其中 N 为系统维数，ne 为 e 中的边缘数，size(g)=[N ne]。对于 Dirichlet 条件的情况，对应值必须为 0。

必须包含每一个边缘上第 1 个点和第 2 个点上的 h 值和 r 值。这样我们有 size(h)=[N^2 2*ne]，其中 N 为系统的维数，ne 为 e 中的边缘个数，且 size(r)=[N 2*ne]。当 M < N 时，h 和 r 必须用 N – M 行 0 来填充。

矩阵 q 和 h 的元素按照 MATLAB 矩阵 q 和 h 中的列序保存。

【例 26-23】　对于边界条件

$$(1-1)u = 2$$

$$\vec{n} \cdot (c \otimes \nabla u) + \begin{pmatrix} 1 & 2 \\ 2 & 0 \end{pmatrix} 4 - \begin{pmatrix} 3 \\ 4 \end{pmatrix} + h'\mu$$

下面的值需要保存在 q，g，h 和 r 中：

```
                1
q=[ ... 2 ... ]
                2
                0
g=[ ... 3 ... ]
                4
```

```
              1    1
h=[ ... 0 ... 0 ... ]
             -1   -1
              0    0
r=[ ... 2 ... 2 ... ]
              0    0
```

2．pdegeom 函数

该函数定义几何 M 文件，其调用格式为：

● ne=pdegeom 通过参数化边缘线段来代表二维区域。区域和边缘线段都被作为标签指定为惟一的正值。边缘线段不能重叠。完整的二维问题的描述应包括几个不相交的区域，它们具有共同的边界线段。区域的边界可以由几个边缘线段组成。所有的边缘线段的连接必须与边缘线段的终点相一致。我们有时候认为一条边缘线段就是一条边界线段或一条内边界线段。边界线段位于最小子域集合的外边界，内边界线段则位于最小子域之间。

有两个选项用于指定问题的几何形态:

（1）用 decsg 函数创建一个离散几何矩阵。可以通过 PDE 图形用户界面（pdetool GUI）自动达到这个目的。使用离散几何矩阵将边缘线段限制为直线、圆弧或椭圆形线段。在工具箱中，离散几何矩阵可以用几何 M 文件代替。

（2）创建一个几何 M 文件。通过创建自己的几何 M 文件，可以创建能精确描述任何数学函数的几何图形。下面的例子演示如何创建 cardioid 函数的图形。

● ne=pdegeom 为边缘线段个数。

● d=pdegeom(bs) 为一矩阵，矩阵中的每一列对应于每一个 bs 中指定的边缘线段。

第 1 列包含初始参数值。

第 2 列包含最终参数值。

第 3 列包含左侧区域的标签。

第 4 列包含右侧区域的标签。

所有子域集合的补集指定为区域个数 0。

● [x,y]=pdegeom(bs,s) 生成边缘线段点的坐标。bs 指定边缘线段，s 指定对应的参数值。bs 可以是一个标量。参数 s 应该与曲线长度近似成比例。所有的最小子域的边界应该至少有两个，最好有 3 个边缘线段。

【例 26-24】 函数 cardg 定义 cardioid 的几何形状。

```
function [x, y]=cardg(bs, s)
```

%CARDG 几何文件，定义 cardioid 的几何形状。

```
nbs=4;
if nargin==0
  x=nbs;
  return
end
dl=[    0       pi/2    pi          3*pi/2
        pi/2    pi      3*pi/2      2*pi;
        1       1       1           1
        0       0       0           0];
if nargin==1
```

```
  x=dl(:, bs);
  return
end
x=zeros(size(s));
y=zeros(size(s));
[m,n]=size(bs);
if m==1  n==1,
  bs=bs*ones(size(s));  expand bs
elseif m =size(s, 1)  n =size(s, 2),
  error('bs must be scalar or of same size as s');
end
nth=400;
th=linspace(0, 2*pi,nth);
r=2*(1+cos(th));
xt=r.*cos(th);
yt=r.*sin(th);
th=pdearcl(th, [xt; yt], s, 0, 2*pi);
r=2*(1+cos(th));
x(:)=r.*cos(th);
y(:)=r.*sin(th);
```

我们用 pdearcl 函数使参数 *s* 与圆弧的长度成比例。键入：

```
pdegplot('cardg'), axis equal
[p, e, t]=initmesh('cardg');
pdemesh(p, e, t), axis equal
```

生成图如图 26-19 所示。

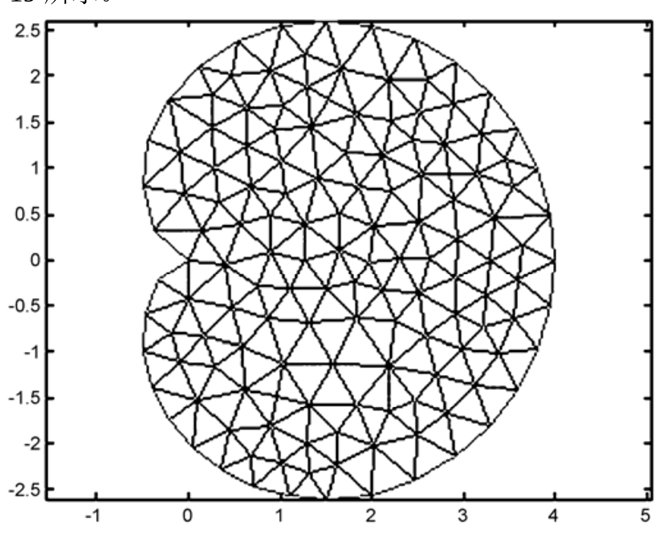


图 26-19　cardioid 的几何形状

然后在 cardioid 定义的几何图形上求解 PDE 问题。在Ω上使用 Dirichlet 边界条件。最后将解用曲面图表示，如图 26-20 所示。

```
u=assempde('cardb',p,e,t,1,0,1);
pdesurf(p,t,u);
```

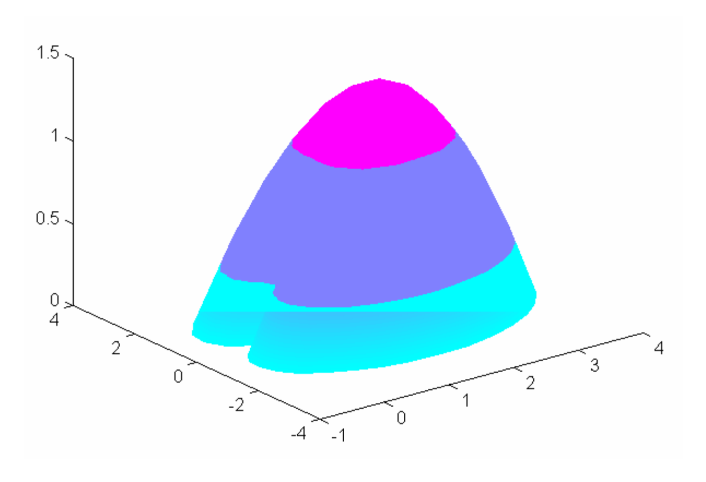


图 26-20　解的图形

第 27 章　利用图形用户界面（GUI）求解偏微分方程的一般过程

MATLAB 提供了一个图形用户界面的偏微分方程数值求解工具。利用该工具，可以交互式地实现偏微分方程数学模型的几何模型建立、边界条件设定、三角形网格剖分和细化、偏微分方程类型设置、参数给定、方程求解和结果图形显示。它包括了数值求解的前处理、计算和后处理等一套完整的程序，可以直观、快速、准确、形象地实现偏微分方程的数值求解。

在 MATLAB 命令窗口中输入命令：pdetool，然后单击回车键，显示 PDE 图形用户界面，如图 27-1 所示。

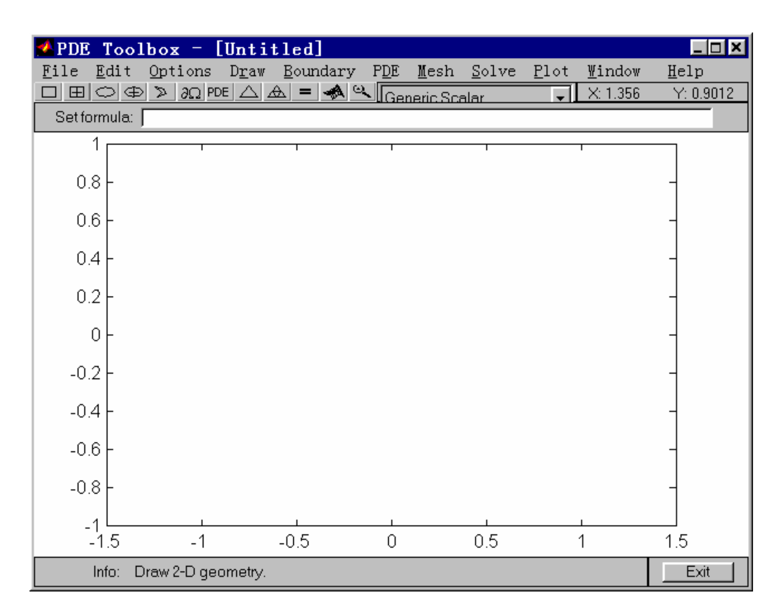


图 27-1　PDE 图形用户界面

下面结合简单实例介绍如何具体使用 PDE 图形用户界面。

一般地，利用 PDE 图形用户界面求解 PDE 问题的过程分为以下几步：

① 选择应用模式；

② 建立几何模型；

③ 定义边界条件；

④ 定义 PDE 类型和 PDE 系数；

⑤ 三角形网格剖分；

⑥ PDE 求解；

⑦ 解的图形表达。

27.1 选择应用模式

PDE 工具箱中根据实际问题的不同提供了很多应用模式，用户可以选择适当的模式进行建模和分析。

在 Options 菜单条中用鼠标指向 Application 选项，会弹出一个子菜单，在其中进行选择，可以确定适当的应用模式。或者直接在工具条中单击 Generic Scalar 下拉式列表框，打开它如图 27-2 所示。在其中进行选择，也可以确定应用模式。列表框中各应用模式的意义分别为：

Generic Scalar
Generic Scalar
Generic System
Structural Mech., Plane Stress
Structural Mech., Plane Strain
Electrostatics
Magnetostatics
AC Power Electromagnetics
Conductive Media DC
Heat Transfer
Diffusion

图 27-2 Generic Scalar 下拉式菜单

- Generic Scalar：一般标量模式（为默认选项）。
- Generic System：一般系统模式。
- Structural Mech.,Plane Stress：结构力学平面应力应用模式。
- Structural Mech.,Plane Strain：结构力学平面应变模式。
- Electrostatics：电静力学应用模式。
- Magnetostatics：磁静力学应用模式。
- AC Power Electromagnetics：交流电电磁学应用模式。
- Conductive Media DC：导电介质直流电应用模式。
- Heat Tranfer：热传导应用模式。
- Diffusion：扩散问题应用模式。

有关各应用模式的具体说明和应用将在后面进行详细介绍。本例中选择默认选项。

27.2 建立几何模型

利用 Draw 菜单中的选项或工具条中的前面五个工具按钮，可以画出 PDE 问题的几何模型。这 5 个工具按钮的意义分别为：

- 绘矩形或方形；
- 绘同心矩形或方形；
- 绘椭圆或圆；
- 绘同心椭圆或圆；
- 绘多义线。

注意，在 Options 菜单中选择 Grid 选项可以在图中显示网格。单击 Grid Spacing…选项，打开 Grid Spacing 对话框，可以调整网格间隔的大小。选择 Snap 选项，则鼠标在图中点击时，将自动选择离该点最近的网格交点。在画图之前设置上面的选项，可以使绘图更方便。

本例中，如图 27-3 所示，在图中画两个同心圆。画好以后，将自动按照先后顺序进行编号。然后在 Set Formula 文本框中输入公式，表示各几何图形之间的集合关系。因为研究域为环形，所以在文本框中输入公式：C1-C2。

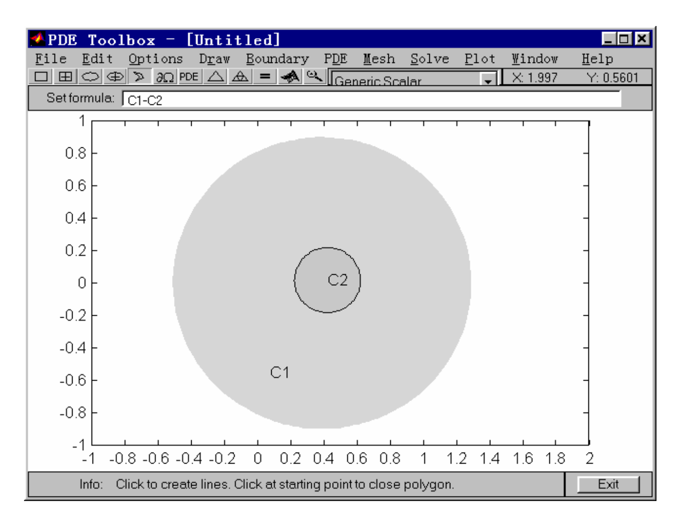


图 27-3 建立几何模型

27.3 定义边界条件

在 Boundary 菜单中选择 Boundary mode 选项或直接在工具条中单击按钮，则显示几何模型的外边界和内边界，如图 27-4 所示。

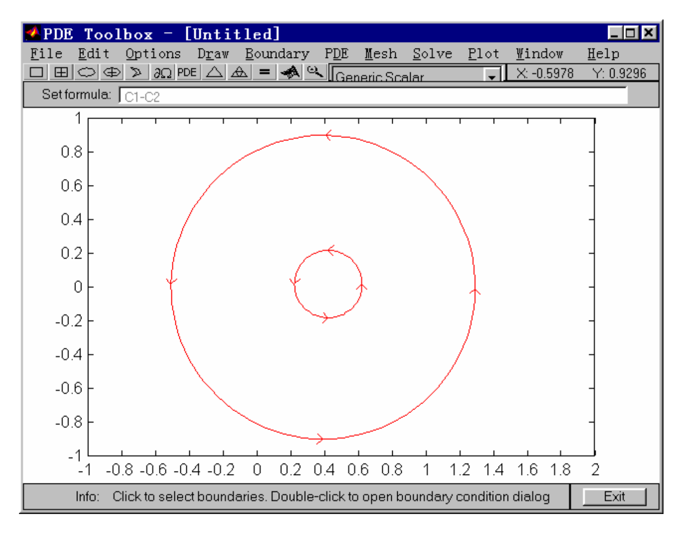


图 27-4 定义边界条件

在图中单击边界，则边界线变成黑色，表示它被选中。按住 Shift 键，可以连续选择多条边界线段。选择要定义的边界以后，双击边界或在 Specify Boundary Conditions…选项，打开“Boundary Condition”对话框，如图 27-5 所示。

对话框中各选项的意义分别为：

- Boundary cond.equation 标签：显示边界条件方程。
- Condition type：在下面的两个单选钮中进行选择，确定边界条件类型。
  - Neumann 单选钮——选择此项，将边界条件类型确定为 Neumann 条件。
  - Dirichlet 单选钮——默认时选择此项。选择此项时，将边界条件类型确定为 Dirichlet

边界条件。

- Coefficient Value 栏：在该栏中的对应文本框中输入边界条件公式中的系数值。本例中选择默认设置。

在 Boundary 菜单中选择 Show Edge Labels 选项和 Show Subdomain Labels 选项，可以在图中显示边缘标签和子域标签。

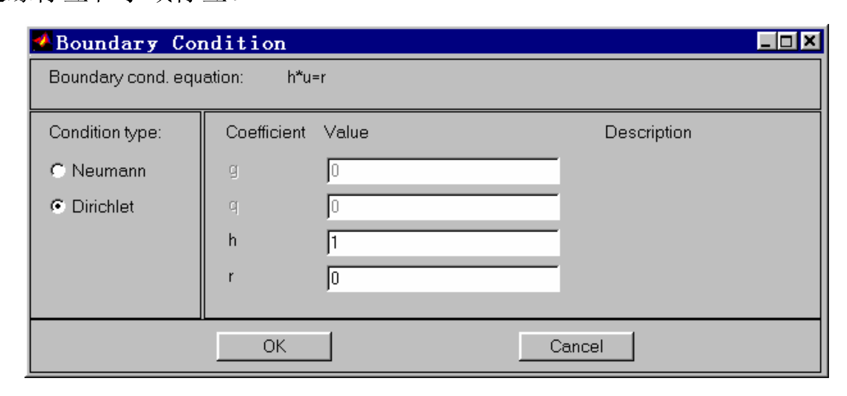


图 27-5　“Boundary Condition”对话框

27.4　定义 PDE 类型和 PDE 系数

在 PDE 菜单中选择 PDE mode 选项，图形将显示为 PDE 模式。选择 Show subdomain labels 选项，将在图中显示子域标签。

在工具条中单击PDE按钮或在 PDE 菜单中单击 PDE Specification...选项，可以打开“PDESpecification”对话框，如图 27-6 所示。

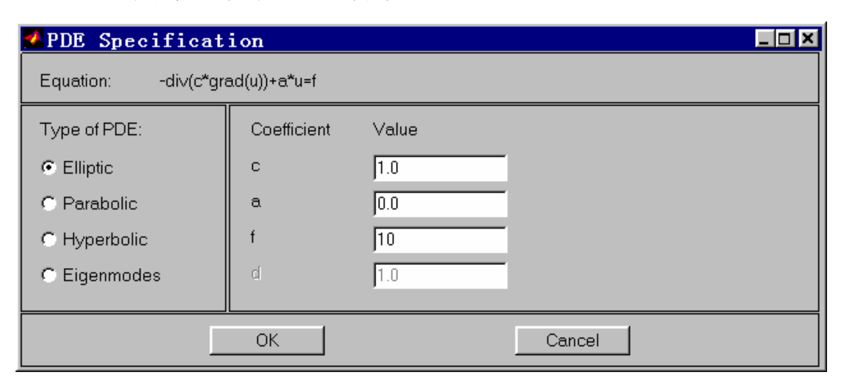


图 27-6　“PDE Specification”对话框

对话框中各选项的意义分别为：

- Equation 标签：该标签显示 PDE 方程表达式。
- Type of PDE 栏：在该栏中进行选择，确定 PDE 问题的类型。
  - Elliptic 单选钮——为默认选项。选择此项，设置为椭圆型 PDE 问题。
  - Parabolic 单选钮——选择此项，设置为抛物线型 PDE 问题。
  - Hyperbolic 单选钮——选择此项，设置为双曲线型 PDE 问题。
  - Eigenmodes 单选钮——选择此项，设置为特征值 PDE 问题。

● Coefficient 栏——在该栏中设置 PDE 方程的系数值。

27.5　三角形网格剖分

在工具栏中单击按钮，或在 Mesh 菜单中选择 Initialize mesh 选项，可以进行研究域的三角形网格初始化，如图 27-7 所示。

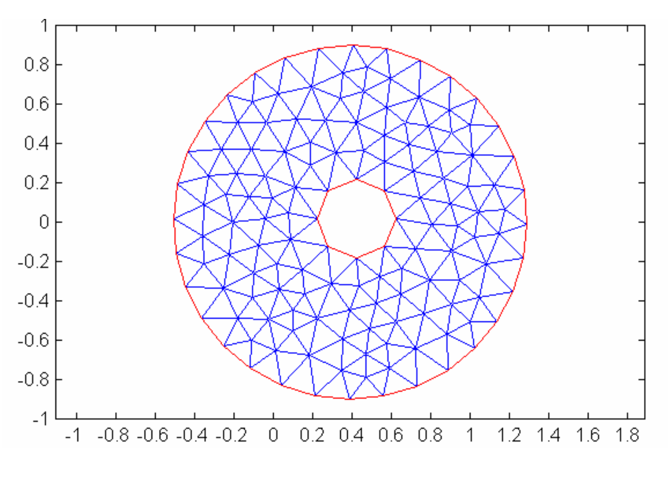


图 27-7　初始化网格

在工具条中单击按钮或在 Mesh 菜单中选择 Refine mesh 选项，可以对初始网格进行细化。如图 27-8 所示。利用细化后的网格进行计算，可以获得具更高精度的解。

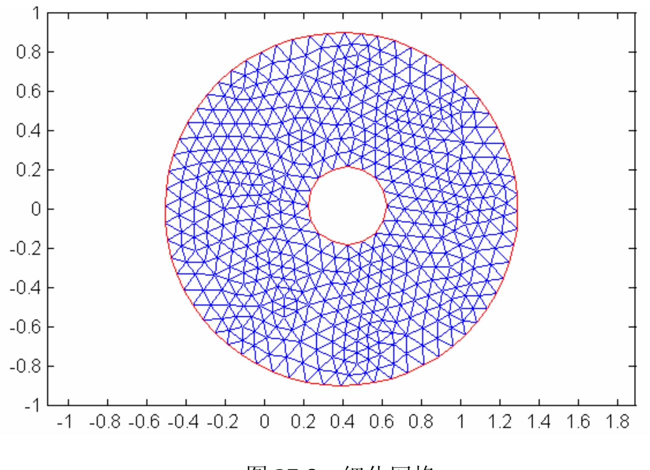


图 27-8　细化网格

在 Mesh 菜单中选择 Jiggle mesh 选项，可以对网格进行微调。选择 Display Triangle Quality 选项，将显示三角形的质量图，如图 27-9 所示。图中，将三角形的最好质量定为 1，并用红色表示，最差质量为 0，用蓝色表示，二者之间的三角形质量根据质量得分的大小用红色和蓝色的过渡色表示。

在 Mesh 菜单中选择 Show Node Labels 选项，将在图中显示网格节点的编号，如图 27-10 所示。选择 Show Subdomain Labels 选项，将显示各子域的编号。

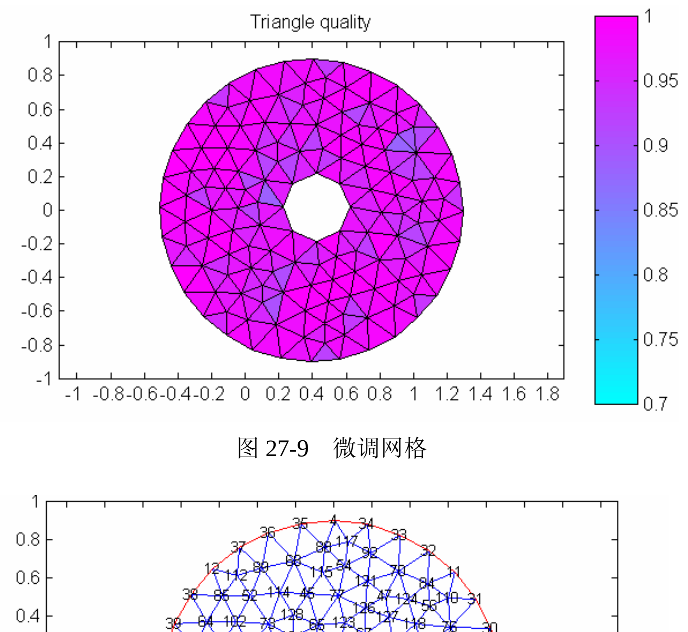


图 27-9　微调网格

图 27-10　显示各子域的编号

在 Mesh 菜单中选择 Parameters…选项，打开“Mesh Parameters”对话框，如图 27-11 所示。在该对话框中进行设置，可以明确与网格剖分有关的参数。

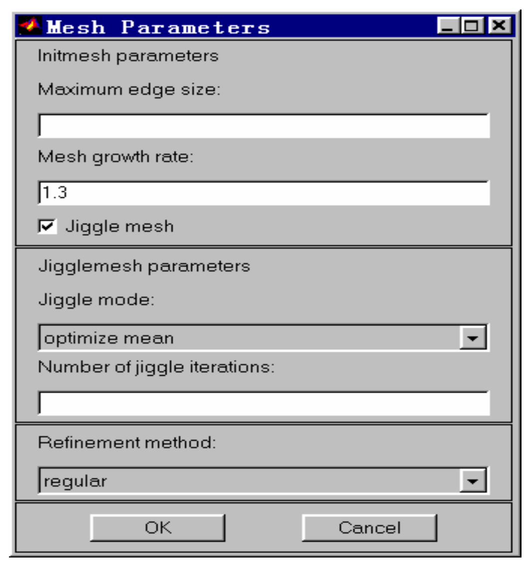


图 27-11　“Mesh Parameters”对话框

对话框中各控件的意义分别为：

- Maximum edge size 文本框：在该文本框中输入数值，确定最大边缘大小。
- Mesh growth rate 文本框：在该文本框中输入网格生长率。默认值为 1.3。
- Jiggle mesh 核选框：选择此项，微调网格。默认时选择此项。
- Jiggle mode 下拉式列表框：在其中进行选择，确定微调模式。可选模式有 on，optimize minimum 和 optimize mean 等 3 项。
- Number of jiggle iterations 文本框：在该文本框中输入微调迭代的次数。
- Refinement method 下拉式列表框：在该控件中进行选择，确定进行网格细化的方法，包括 regular 和 longest 两个选项。具体内容参见前面函数部分。

27.6 PDE 求解

在工具条中单击按钮或在 Solve 菜单中选择 Solve PDE 选项，可以对前面定义的 PDE 问题进行求解。本例的解如图 27-12 所示。

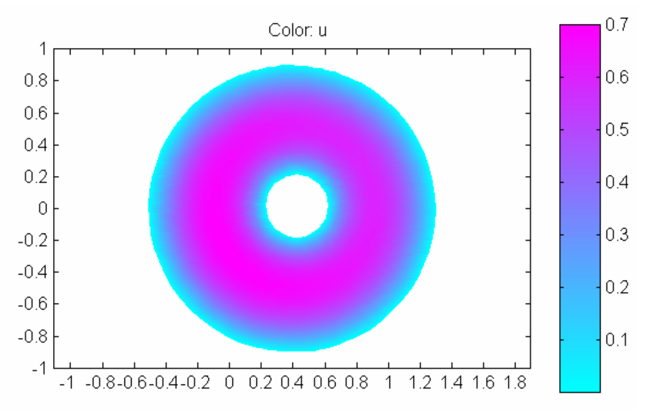


图 27-12　问题的默认图解（色谱图）

在 Solve 菜单中单击 Parameters…选项，打开“Solve Parameters”对话框，如图 27-13 所示。在该对话框中进行设置，可以确定求解方法和参数。

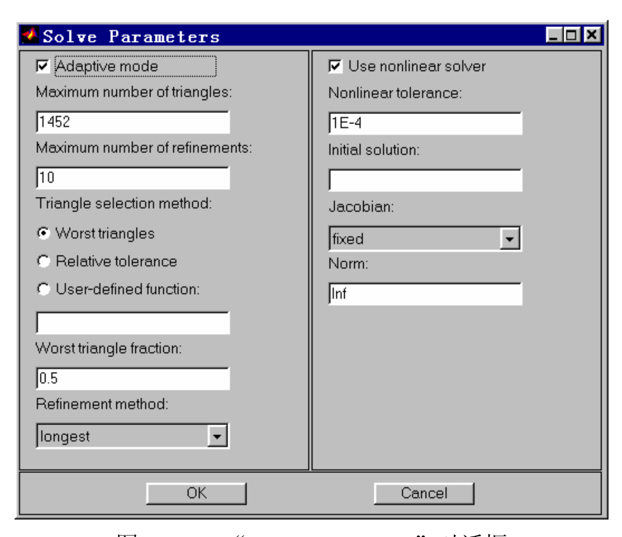


图 27-13　“Solve Parameters”对话框

该对话框中各选项的意义分别为：

● Adaptive mode 核选框：选择此项，即选择自适应模式，系统将自适应生成网格并进行求解。

● Maximum number of triangles 文本框：在其中输入三角形的最大个数。

● Maximum number of refinements 文本框：在其中输入网格细化的最大次数。

● Triangle selection method：在下面的 3 个单选钮中进行选择，确定三角形的选择方法。

➢ Worst triangles 单选钮——为默认选项。选择此项，根据最坏三角形进行选择。

➢ Relate tolerance 单选钮——选择此项，根据相对容限进行选择。

➢ User-defined function 单选钮——选择此项，在下面的文本框中输入函数，用该函数选择三角形。

● Worst triangle fractions 文本框：在该文本框中输入最坏三角形分量。

● Refinement method 下拉式列表框：在其中进行选择，确定进行网格细化的方法，有 regular 和 longest 两个选项。

● User nonlinear solver 核选框：选择此项，用非线性求解器进行求解。

● Nonlinear tolerance 文本框：在该文本框中输入非线性迭代终止的容限。默认值为 1E-4。

● Initial solution 文本框：在该文本框中输入问题的解的初值。

● Jacobian 下拉式列表框：在该控件中进行选择，确定雅可比矩阵的确定方式。有 fixed, lumped 和 full 3 个选项。

● Norm 文本框：在其中输入范数值，默认值为 Inf（∞）。

27.7 解的图形表达

PDE 工具项提供解的多种图形表达方式。上面显示的是彩色图，为默认显示。单击按钮或在 Solve 菜单中单击 Parameters…选项，可以打开“Plot Selection”对话框，如图 27-14 所示。

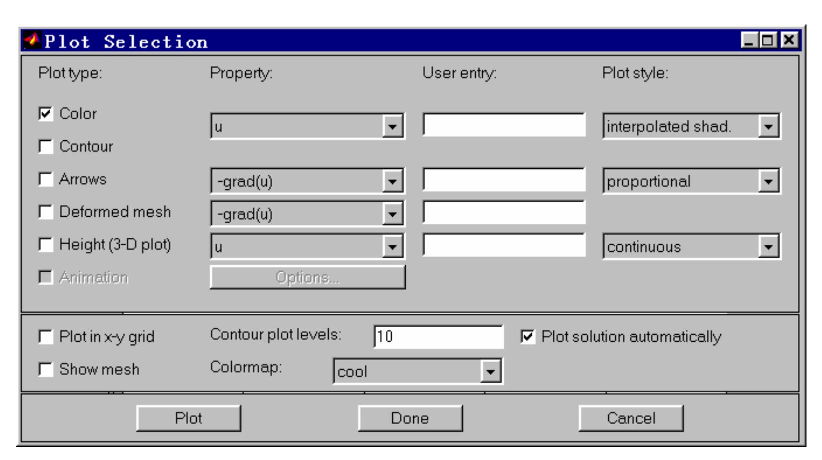


图 27-14 “Plot Selection”对话框

该对话框中各选项的意义分别为：

● Plot type 控件列：该列控件控制图形类型的选择。包括：

➢ Color 核选框——选择此项，生成并显示解的彩色图。默认时选择此项。

➢ Contour 核选框——选择此项，生成并显示解的等值线图，如图 27-15 所示。

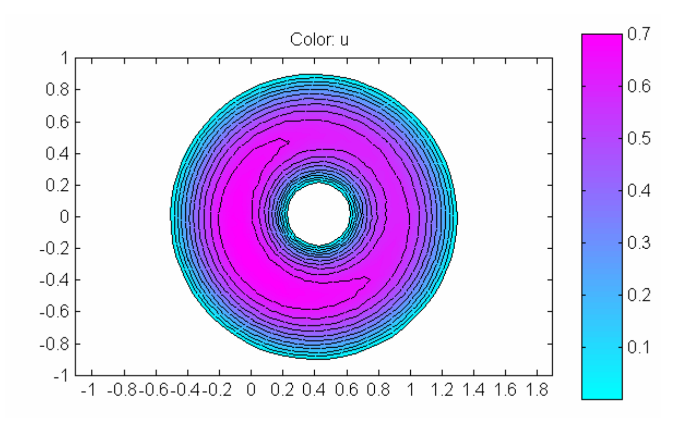


图 27-15　色谱图叠加等值线图

➢ Arrows 核选框——选择此项，生成并显示解的矢量图，如图 27-16 所示。

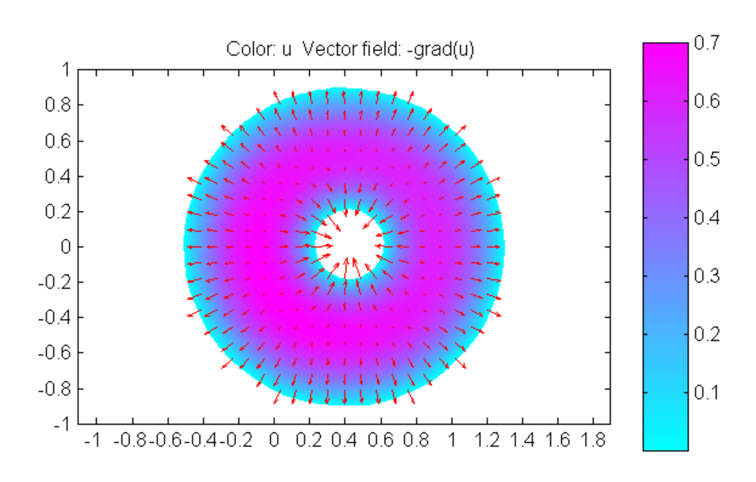


图 27-16　色谱图叠加矢量图

➢ Deformed mesh 核选框——选择此项，生成并显示解的变形网格图，如图 27-17 所示。

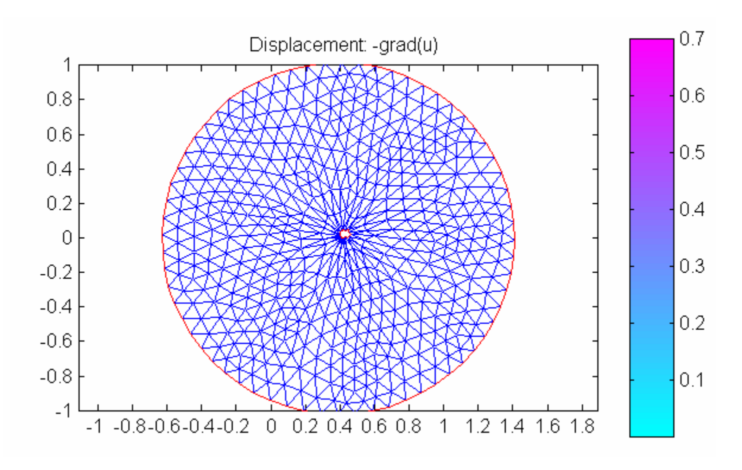


图 27-17　变形网格图

➢ Height(3-D plot)核选框——选择此项，生成并显示解的三维图，如图 27-18 所示。

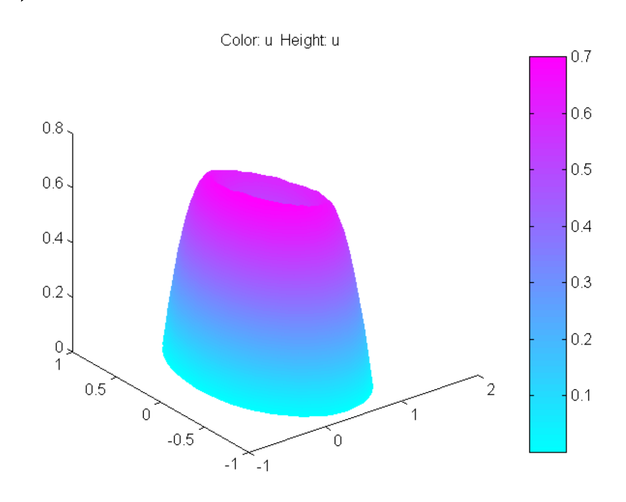


图 27-18　三维高程图

➢ Animation 核选框——选择此项，生成解的系列演示图。

- Property 控件列——该控件列为一组下拉式列表框，定义对解的哪一部分进行图形显示。
- User entry 控件列：为一组文本框，在其中输入用户输入。
- Plot style 控件列：该控件列控制前面选择图型的不同风格。
- Plot in x-y grid 核选框：选择此项，在 x-y 网格中绘图。
- Show mesh 核选框：选择此项，在当前图中显示网格。
- Contour plot levels 文本框：在其中输入等值线的水平数。
- Colormap 下拉式列表框：在该控件中选择绘彩色图的颜色。
- Plot solution automatically 核选框：选择此项，系统自动绘制解的图形。

第 28 章　几种常见的偏微分方程数值求解问题

28.1　椭圆型问题

28.1.1　单位圆盘的泊松方程

泊松方程是最简单的椭圆型 PDE 问题。

该问题的公式为

$$-\Delta U = 1$$

边界上 U=0。该问题的精确解为

$$U(x,y) = \frac{1 - x^2 - y^2}{4}$$

1．用图形用户界面计算

在命令窗口中输入 pdetool 命令，选用 Generic Scalar 模式

① 单击 Option 菜单，选择 add a grid 选项，设置“snap-to-grid”特点。单击按钮画一个圆。若该图不是标准的单位圆，则双击该圆，打开一对话框，在其中可以指定圆心的精确位置和半径的大小。

② 通过单击按钮来设置边界模式。分割的几何边界显示出来，并且外边界指定为默认设置，即 Dirichlet 边界条件，u=0。本例中采用默认设置，若边界条件不同，可以通过双击边界打开一对话框，在其中输入对应的边界条件。

③ 单击按钮，定义偏微分方程，该操作打开一对话框，可以在其中定义 PDE 系数 c,a 和 f。本例中，它们均为常数：c=1, f=1, a=0。

④ 单击按钮或选择 Mesh 菜单中的 Initialize Mesh 选项，初始化显示三角形网格。

⑤ 单击按钮或在 Mesh 菜单中选择 Refine Mesh 选项，改进初始网格并显示新网格，如图 28-1 所示。

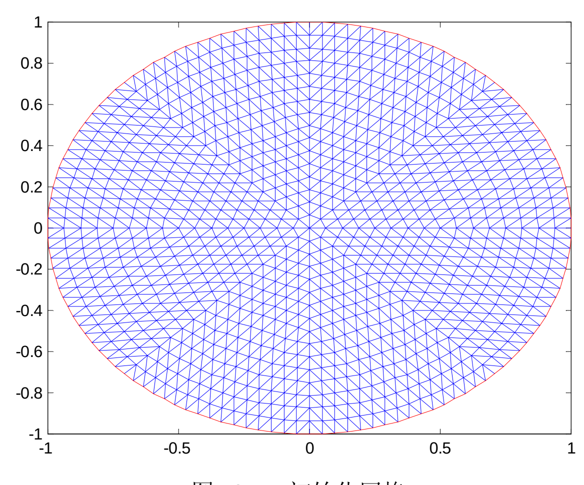


图 28-1　初始化网格

⑥ 单击 按钮进行求解，MATLAB 可以用图形来表示问题的解。单击 按钮，打开 Plot Selection 对话框。利用该对话框，可以选择不同类型的解图。默认时本问题解的色谱图如图 28-2 所示。

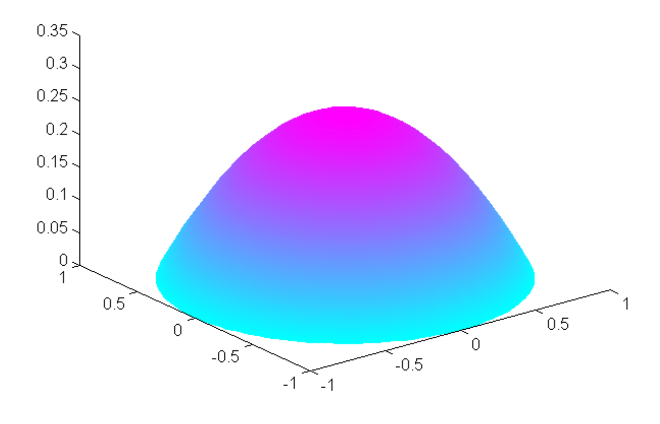


图 28-2　问题解的色谱图

⑦ 比较数值解与精确解之间的误差。

为 Plot Selecting 对话框中的 Color 选择 Property 弹出式菜单中的 User entry。然后在 User entry 编辑区中输入 MATLAB 表达式 u-(1-x.^2-y.^2)/4。可以获得解的绝对误差的图形表示。如图 28-3 所示。

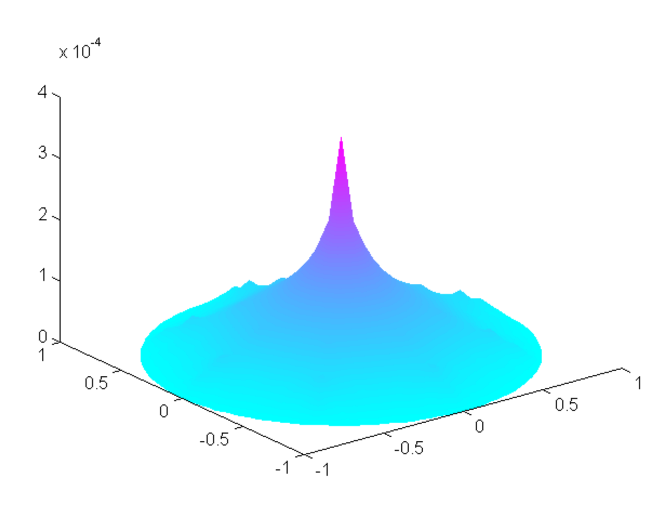


图 28-3　解的绝对误差的表面图形

2．使用命令行函数

首先必须创建 MATLAB 函数，使二维几何模型参数化。

M 文件 circleg.m 返回单位圆盘边界点的坐标。下面是该文件的内容：

```
nbs=4;
if nargin==0,
  x=nbs;     %边界线段个数
  return
end
```

```
d=[
  0 0 0 0       % 参数初值
  1 1 1 1       % 参数终值
  1 1 1 1       % 左端区域
  0 0 0 0       % 右端区域
];

bs1=bs(:)';

if find(bs1<1 | bs1>nbs),
  error('Non existent boundary segment number')
end

if nargin==1,
  x=d(:, bs1);
  return
end

x=zeros(size(s));
y=zeros(size(s));
[m,n]=size(bs);
if m==1 & n==1,
  bs=bs*ones(size(s));   % 扩展 bs
elseif m~=size(s, 1) | n~=size(s, 2),
  error('bs must be scalar or of same size as s');
end

if ~isempty(s),

% 边界线段 1
ii=find(bs==1);
x(ii)=1*cos((pi/2)*s(ii)-pi);
y(ii)=1*sin((pi/2)*s(ii)-pi);

% 边界线段 2
ii=find(bs==2);
x(ii)=1*cos((pi/2)*s(ii)-(pi/2));
y(ii)=1*sin((pi/2)*s(ii)-(pi/2));

% 边界线段 3
ii=find(bs==3);
x(ii)=1*cos((pi/2)*s(ii));
y(ii)=1*sin((pi/2)*s(ii));

% 边界线段 4
ii=find(bs==4);
x(ii)=1*cos((pi/2)*s(ii)-(3*pi/2));
```

```
y(ii)=1*sin((pi/2)*s(ii)-(3*pi/2));

end
```

然后用另一函数 circleb1.m 描述边界条件。

```
function [q, g, h, r]=circleb1(p, e, u, time)
bl=[
    1 1 1 1
    1 1 1 1
    1 1 1 1
    1 1 1 1
    1 1 1 1
    1 1 1 1
    48 48 48 48
    48 48 48 48
    49 49 49 49
    48 48 48 48
  ];

if any(size(u))
  [q, g, h, r]=pdeexpd(p, e, u, time, bl);
else
  [q, g, h, r]=pdeexpd(p, e, time, bl);
end
```

现在可以用命令行进行工作：

```
[p, e, t]=initmesh('circleg', 'Hmax', 1);
error=[ ];err=1;
while err>0.001,
      [p, e, t]=refinemesh('circleg', p, e, t);
      u=assempde('circleb1', p, e, t, 1, 0, 1);
      exact=-(p(1,:).^2+p(2,:).^2-1)/4;
      err=norm(u-exact', inf);
      error=[error, err];
   end
pdemesh(p, e, t)
```

生成网格图（见图 28-1）。

```
pdesurf(p, t, u)
```

生成解的色谱图（见图 28-2）。

```
pdesurf(p, t, u-exact')
```

生成解的绝对误差的表面图（见图 28-3）。

上面的命令行中，第 1 行用参数化函数 circleg 创建初始网格。

为解的最大误差设定初始向量 error，设置初始误差 err 为 1。下面的循环将一直进行下去，直到解的误差小于 10^{-3}。

① 改进网格；

② 求解线性系统，注意本例中椭圆型 PDE 的系数为常数（c=f=1, a=0）,circleb1 包含边界条件的描述，p, e, t 定义三角形网格；

③ 求数值解与精确解的误差。Exact 向量包含节点处的精确解；

④ 绘网格、解和误差的图形；

28.1.2　一个离散问题

本例计算从被瞬间照亮的物体上反射回来的波。此问题可概括为一张周边固定在物体上的无限水平的薄膜产生了小的垂向变形 U。

假设介质为均质的，则波速为常数 c。

当时间段内照明光线是和谐一致的，则可以通过解简单的稳定问题来计算区域。

$$\frac{\partial^2 U}{\partial t^2} - c^2 \Delta U = 0$$

由于 $U(x,y,t) = u(x,y)e^{-i\omega t}$，波动方程变为 $-\omega^2 u - c^2 \Delta u = 0$ 或 Helmholtz 方程 $-\Delta u - k^2 u = 0$，其中 k 为波数，与角频率(angular frequency)ω、频率 f 和波长λ 有关。即

$$k = \frac{\omega}{c} = \frac{2\pi f}{c} = \frac{2\pi}{\lambda}$$

下面指定边界条件，假设瞬时波是方向为 $\bar{a} = (\cos(a)\sin(a))$ 的平面波，则有

$$V(x,y,t) = \mathrm{e}^{i(k\bar{a}\cdot\bar{x} - \omega t)} = v(x,y)\mathrm{e}^{-i\omega t}$$

其中，

$$v(x,y) = \mathrm{e}^{ik\bar{a}\cdot\bar{x}}$$

u 为 V 和 r（反射波）之和

$$u = V + r\ u$$
$$r = -v(x,y)$$

物体边界的条件很简单：u=0。

r 近似满足单向波动方程。

它允许波只在ξ 的正向上移动（ξ 为源于物体的发射距离）。由于是时间和谐解，它可以转化为广义 Neumann 边界条件

$$\bar{\xi} \cdot \nabla r = ikr$$

$$\frac{\partial r}{\partial t} + c\bar{\xi} \cdot \nabla r = 0$$

1．用 GUI

输入命令行 pdetool，然后在窗口界面中进行以下操作：

① 选择 Generic Scalar 模式。

② 绘制二维几何图形。令被照射的物体为一边宽为 0.1 单位的方形 SQ1，其中心位置为[0.8, 0.5]，旋转 45 度。计算域为半径为 0.45 的圆，中心位置与方形相同。

③ 设置边界。对于外边界的设置，边界条件为广义 Neumann 条件：q=-ik，波的个数为 k=60，对应于 0.1 单位的波长，故输入 q=-60i，g=0。方形物体的边界必须满足 Dirichlet 条件。本例中，偶然波在-x 方向上传播，故边界条件为

$$r = -v(x,y) = -\mathrm{e}^{ik\bar{a}\cdot\bar{x}}$$

将此边界条件输入到 Boundary Condition 对话框中，作为 Dirichlet 条件：

$$r = -\mathrm{e}^{-ikx}$$

$$h=1,\ r=-\exp(-i*60*x)$$

本问题的椭圆型 PDE 系数为：c=1, a=–k^2=–3600, f=0。在 PDE Specification 对话框中输入这些值，然后求解。

反射波的衍射由下式计算

$$\mathrm{Re}(r(x,y)\mathrm{e}^{-k\omega t})$$

该式表示下面公式所代表的波的反射。

$$\mathrm{Re}(\mathrm{e}^{i(k\bar{a}\cdot\bar{x})-\omega t})$$

本问题的几何模型网格如图 28-4 所示。解的图形如图 28-5 所示。

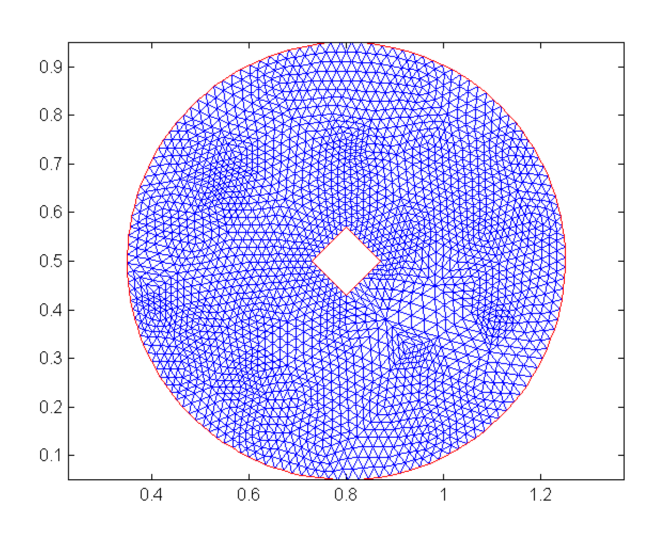


图 28-4　模型网格

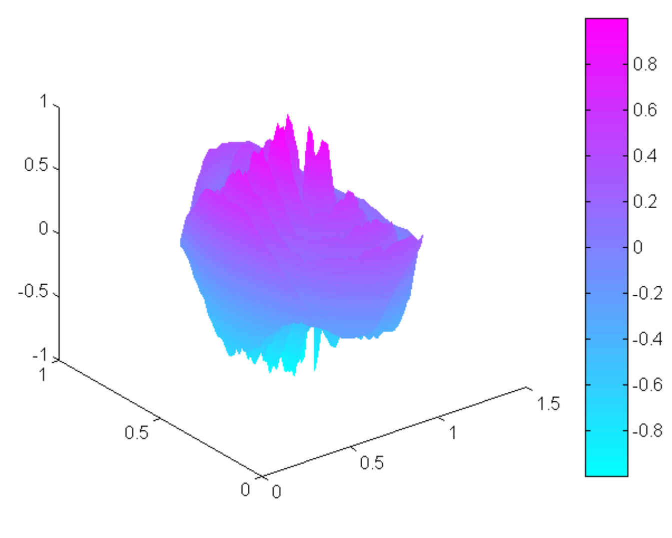


图 28-5　解的三维图

2．使用命令行函数

在命令行中键入

```
k=60;
```

```
g='scatterg';      % 带矩形窗口的圆形
b='scatterb';      % 边界条件
c=1;
a=-k^2;
f=0;

%网格剖分、细化
[p, e ,t]=initmesh(g);
[p,e,t]=refinemesh(g, p, e, t);
[p, e, t]=refinemesh(g, p, e, t);
%绘制网格图
pdemesh(p, e, t); axis equal
u=assempde(b, p, e, t, c, a, f);

h = newplot; set(get(h, 'Parent'), 'Renderer', 'zbuffer')
pdeplot(p, e, t, 'xydata', real(u), 'zdata', real(u), 'mesh', 'off');
colormap(cool)
```

然后编制 M 文件或直接以命令行的形式输入以下命令：

```
h=newplot; hf=get(h, ’Parent’); set(hf, ’Renderer’, ’zbuffer’)
axis tight, set(gca, ’DataAspectRatio’, [1 1 1]); axis off
M=moviein(10, hf);
maxu=max(abs(u));
colormap(cool)
for j=1:10,
    ur=real(exp(-j*2*pi/10*sqrt(-1))*u);
    pdeplot(p, e, t, ’xydata’, ur, ’colorbar’, ’off’, ’mesh’, ’off’);
    caxis([-maxu maxu]);
    axis tight, set(gca, ’DataAspectRatio’, [1 1 1]); axis off
    M(:, j)=getframe;
  End
movie(hf, m, 50);
```

生成如图 28-6 所示的快照图形。movie 函数用动画的方式展示了波的传播过程。

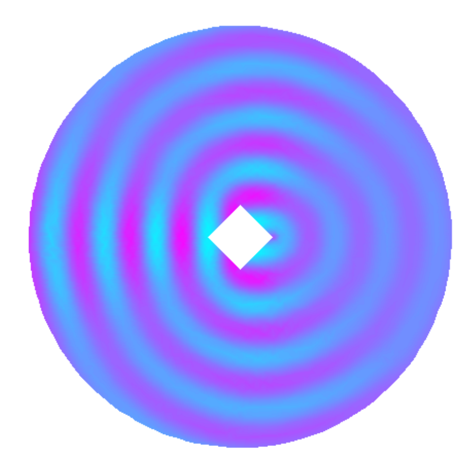


图 28-6　波的传播快照图

28.1.3　最小表面问题

在许多问题中，系数 c、a 和 f 不仅仅取决于 x 和 y，也与解 u 本身有关。考虑下面的方程

$$-\nabla\left(\frac{1}{\sqrt{1+\left|\nabla u\right|^2}}\right)=0$$

圆盘域为 $\Omega=\{(x, y)\,|\,x^2+y^2\leqslant 1\}$，边界条件为 $u=x^2$。

本问题为非线性问题，不能用常规的椭圆型方程求解器进行求解，而需要用非线性求解器 pdenonlin。

1．使用 GUI

在命令窗口输入命令 pdetool，打开 PDE 工具，选择 Generic Scalar 应用模式。

问题子域为单位圆，画出单位圆，移到 Boundary 模式，定义边界条件。在 Edit 菜单条中选择 Select All 选项，选择所有边界。然后双击边界，打开 Boundary Condition 对话框。在 r 编辑框中输入 x.^2，定义 Dirichlet 条件 u=x2。然后打开 PDE Specification 对话框，定义 PDE。这是一个椭圆型方程，表达式为

$$c=-\nabla\left(\frac{1}{\sqrt{1+\left|\nabla u\right|^2}}\right), a=0, f=0$$

在 c 编辑框中输入：

```
1./sqrt(1+ux.^2+uy.^2)
```

将网格初始化并细化一次。

求解 PDE 以前，选择 Solve—Parameters…，核对 Use nonlinear solver 单选钮，将容限参数设为 0.001。

单击 = 按钮，进行求解。

在 Plot Selection 对话框中进行设置，绘制 3-D 解。

2．用命令行函数

circlegM 文件和 circleb2.m 分别包含模型的几何参数和边界条件函数。

```
g='circleg';
b='circleb2';
c='1./sqrt(1+ux.^2+uy.^2)';
rtol=1e-3;

[p, e, t]=initmesh(g);
[p, e, t]=refinemesh(g, p, e, t);

u=pdenonlin(b, p, e, t, c, 0, 0, 'Tol', 'Tol', rtol);

pdesurf(p, t, u)
```

结果如图 28-7 所示。

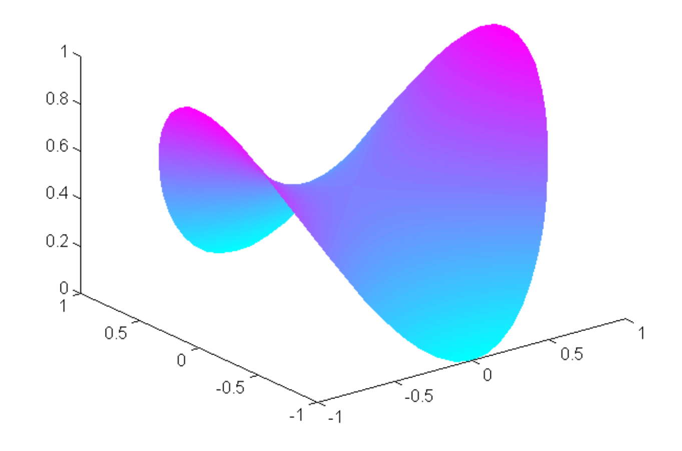


图 28-7　最小表面问题的解

28.1.4　区域分解问题

若研究域具有复杂的几何边界，常将它分解为结构更为简单的子域。

假设 Ω 由子域 Ω1, Ω2,…, Ω*n* 组成，可以重新为网格上的节点编号，这样每个子域上节点的系数被分到同一组。刚度矩阵 ***K*** 为

$$\boldsymbol{K}=\begin{bmatrix}\boldsymbol{K}_1 & \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{B}_1^T\\ \mathbf{0} & \boldsymbol{K}_2 & \cdots & \mathbf{0} & \boldsymbol{B}_2^T\\ \vdots & \vdots & \ddots & \vdots & \vdots\\ \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{K}_n & \boldsymbol{B}_n^T\\ \boldsymbol{B}_1 & \boldsymbol{B}_2 & \cdots & \boldsymbol{B}_n & \boldsymbol{C}\end{bmatrix}$$

右端项为：

$$\boldsymbol{F}=\begin{bmatrix}\boldsymbol{f}_1\\ \boldsymbol{f}_2\\ \vdots\\ \boldsymbol{f}_n\\ \boldsymbol{f}_c\end{bmatrix}$$

PDE 工具箱过程 assempde 可以集合矩阵 $\boldsymbol{K}_j$，$\boldsymbol{B}_j$，$\boldsymbol{f}_j$ 和 $\boldsymbol{C}$。

线性系统的结构为

$$\boldsymbol{K}u=\boldsymbol{F}$$

由于将 ***K*** 分解到上面的部分矩阵而得到简化。

考虑 L 形薄膜的几何形状，可以通过下面的命令行来生成几何图。

```
pdegplot('lshapeg')
```

注意到子域之间的边界，有 3 个子域，这样 *n*=3。现在生成几何图形的网格：

```
[p, e, t]=initmesh('lshapeg');
[p, e, t]=refinemesh('lshapeg', p, e, t);
[p, e, t]=refinemesh('lshapeg', p, e, t);
```

本例中，*n*=3，有

$$\begin{bmatrix} \boldsymbol{K}_1 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{B}_1^T \\ \boldsymbol{0} & \boldsymbol{K}_2 & \boldsymbol{0} & \boldsymbol{B}_2^T \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{K}_3 & \boldsymbol{B}_3^T \\ \boldsymbol{B}_1 & \boldsymbol{B}_2 & \boldsymbol{B}_3 & \boldsymbol{C} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \\ \boldsymbol{u}_3 \\ \boldsymbol{u}_4 \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \\ \boldsymbol{f}_3 \\ \boldsymbol{f}_4 \end{bmatrix}$$

解为

$$(\boldsymbol{C} - \boldsymbol{B}_1 \boldsymbol{K}_1^{-1} \boldsymbol{B}_1^T - \boldsymbol{B}_2 \boldsymbol{K}_2^{-1} \boldsymbol{B}_2^T - \boldsymbol{B}_3 \boldsymbol{K}_3^{-1} \boldsymbol{B}_3^T) \boldsymbol{u}_c = \boldsymbol{f}_c - \boldsymbol{B}_1 \boldsymbol{K}_1^{-1} \boldsymbol{f}_1 - \boldsymbol{B}_2 \boldsymbol{K}_2^{-1} \boldsymbol{f}_2 - \boldsymbol{B}_3 \boldsymbol{K}_3^{-1} \boldsymbol{f}_3$$

$$\boldsymbol{u}_1 = \boldsymbol{K}_1^{-1} (\boldsymbol{f}_1 - \boldsymbol{B}_1^T \boldsymbol{u}_c)$$

$$\cdots$$

下面列出的是一个更有效的算法，它用到乔累斯基因子提取。

```
time=[ ];
np=size(p, 2);
%首先找到公共点
cp=pdesdp(p, e, t);

%分配空间
nc=length(cp);
C=zeros(nc, nc);
FC=zeros(nc, 1);
Pause    % 单击任意键继续

% 组合子域 1 并进行更新
[i1, c1]=pdesdp(p, e, t, 1);ic1=pdesubix(cp, c1);
[K, F]=assempde(b, p, e, t, c, a, f, time, 1);
K1=K(i1, i1); d=symmmd(K1); i1=i1(d);
K1=chol(K1(d, d)); B1=K(c1, i1); a1=B1/K1;
C(ic1, ic1)=C(ic1, ic1)+K(c1, c1)-a1*a1';
f1=F(i1); e1=K1'\f1; FC(ic1)=FC(ic1)+F(c1)-a1*e1;
pause % 单击任意键继续

% 组合子域 2 并进行更新
[i2, c2]=pdesdp(p, e, t, 2); ic2=pdesubix(cp, c2);
[K, F]=assempde(b, p, e, t, c, a, f, time, 2);
K2=K(i2, i2); d=symmmd(K2); i2=i2(d);
K2=chol(K2(d, d)); B2=K(c2, i2); a2=B2/K2;
C(ic2, ic2)=C(ic2, ic2)+K(c2, c2)-a2*a2';
f2=F(i2); e2=K2'\f2; FC(ic2)=FC(ic2)+F(c2)-a2*e2;
pause    %单击任意键继续

% 组合子域 3 并进行更新
[i3, c3]=pdesdp(p, e, t, 3); ic3=pdesubix(cp, c3);
[K, F]=assempde(b, p, e, t, c, a, f, time, 3);
K3=K(i3, i3); d=symmmd(K3); i3=i3(d);
K3=chol(K3(d, d)); B3=K(c3, i3); a3=B3/K3;
C(ic3, ic3)=C(ic3, ic3)+K(c3, c3)-a3*a3';
f3=F(i3); e3=K3'\f3; FC(ic3)=FC(ic3)+F(c3)-a3*e3;

%求解
u=zeros(np, 1);
u(cp)=C\FC; % Common points
u(i1)=K1\(e1-a1'*u(c1)); % Points in SD 1
u(i2)=K2\(e2-a2'*u(c2)); % Points in SD 2
```

```
u(i3)=K3\(e3-a3'*u(c3)); % Points in SD 3

% 绘图
pdesurf(p, t, u)
```

结果如图 28-8 所示。

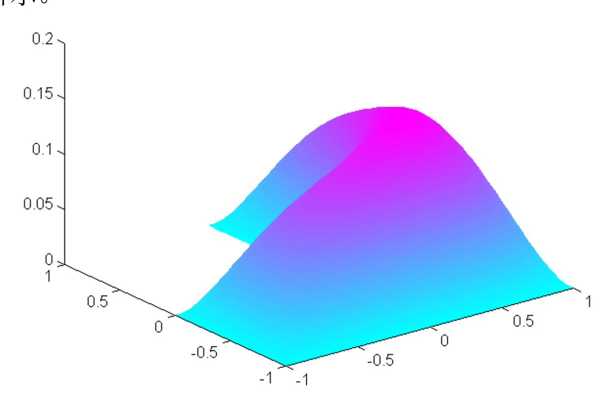


图 28-8　L 形薄膜上的解

也可以通过输入以下命令行来求解问题：

```
%与不同子域分解法得到的解相比。
[K, F]=assemble(‘lshapeb’, p, e, t, 1, 0, 1);
u1=K\F;
norm(u-u1, ’inf’)
pdesurf(p, t, u)
```

28.2　抛物线型问题

28.2.1　受热金属块的热传导方程

常见的抛物线型问题可以用热传导方程来表达，即

$$d\frac{\partial u}{\partial t}-\Delta u=0$$

它描述某些物体的热扩散发程。

本例研究一块受热的有矩形裂纹的金属块。金属块的左侧被加热到 100℃，在右侧热量则以恒定速率降低到周围空气的温度，所有其他边界都是独立的，于是引出下列边界条件：

（1）u=100　　　　左侧（Dirichlet 条件）

（2）$\frac{\partial u}{\partial n}=-10$　　　　右侧(Neumann 条件)

（3）$\frac{\partial u}{\partial n}=0$　　　　其他边界(Neumann 条件)

对于热传导方程来说，还需要一个初值，即起始时间 t_0 时金属块的温度。本例中，金属块的初始温度为 0℃。

最后，完成问题表达式。指定起始时间为 0，而且希望研究开始 5 秒钟的热扩散问题。

1．用 GUI

打开图形用户界面，画 CSG 模型。

首先画一个矩形（R1），其四角的坐标为 x=[-0.5 0.5 0.5 –0.5]，y=[-0.8 –0.8 0.8 0.8]。然后画另一个矩形（R2），代表矩形裂纹，它的四角坐标为 x=[-0.05 0.05 0.05 –0.05]，y=[-0.4 –0.4 0.4 0.4]。选择 Options 选项，打开 Grid Spacing 对话框，在-0.05 和 0.05 处输入 X 轴的附加短线，以帮助画出代表裂纹的矩形。然后显示网格和“Snap-to-grid”风格，这样就很容易画出矩形裂纹了。

金属块的 CSG 模型现在可用下面的简单公式来表示：R1-R2。

单击∂Ω按钮，输入 Boundary 模式，选择边界并指定边界条件。

单击PDE按钮，打开 PDE Specification 对话框，输入 PDE 系数。

需要解决的 PDE 表达式为

$$d\frac{\partial u}{\partial t}-\nabla\cdot(c\nabla u)+au=f$$

初值 u0=u(t0)，计算的时间保存在数组 tlrst 中。

本例中，d=1，c=1，a=0，f=0。

单击△按钮，将网格初始化，单击△按钮改进网格。

初值 u0=0，时间用[0 : 0.5 : 5]的形式输入到 Solve Parameters 对话框（通过在 Solve 菜单条中选择 Parameters…选项来打开）中。

单击=按钮，在 11 个不同时间段内求解热传导方程，默认时，显示最后时间解的 interpolated 图。

另外有一个更有趣的动态显示热传导过程的方法，首先在 Plot Selection 对话框中选择 Animation 核选框，选择 colormap hot，单击 Plot 按钮，在单独的图形窗口中开始解的图形记录。然后会重复照亮 5 次。

注意到金属块的温度升高很快，为改进照明效果并注重第 1 秒钟，可试着改变时间列表的表达式 logspace(-2, 0.5, 20)。

也可以试着改变热能系数 d 和右边界的热流量来研究它们是如何影响热扩散的。

2. 用命令行函数

首先建立几何模型和边界条件的 M 文件。金属块的几何模型在 M 文件 crackg.m 中，边界条件保存在 crackb.m 中。

创建初始网格，调用函数 initmesh。

```
[p, e, t]=initmesh('crackg');
```

然后可以用函数 Parabolic 求解热传导方程。

```
u=parabolic(0,0:0.5:5,'crackb',p,e,t,1,0,0,1);
Time: 0.5
Time: 1
Time: 1.5
Time: 2
Time: 2.5
Time: 3
Time: 3.5
Time: 4
Time: 4.5
Time: 5
153 successful steps
0 failed attempts
308 function evaluations
1 partial derivatives
28 LU decompositions
```

```
307 solutions of linear systems
```

得到的解 u 是一个有 11 列的矩阵，其中每一列对应于 11 个时间点上的解。

可用 pdeplot 函数生成不同时间点上解的图形，如 t=5.0 时，

```
pdeplot(p, e, t, 'xydata', u(:, 11), 'mesh', 'off', 'colormap', 'hot')
```

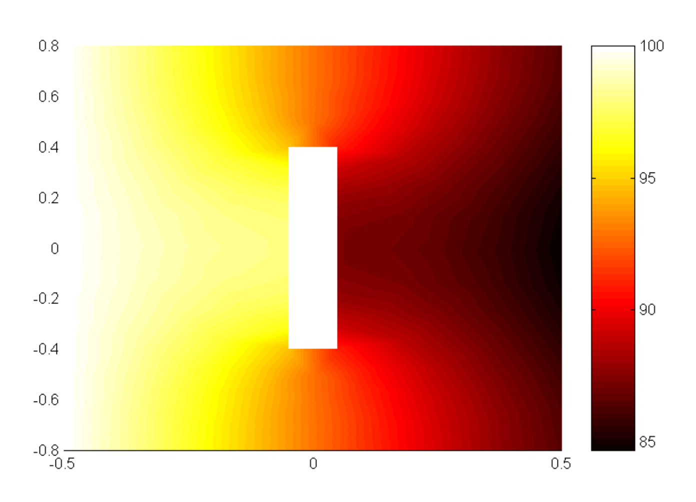


图 28-9　金属块热传导问题的解

28.2.2　放射性棒的热扩散

考察一根放射性棒，在其左端持续不断地加以热源，右端恒温。在外边界上，棒体与外界发生热交换。同时，由于放射性过程，整个棒体均匀地产生热量。假设初始温度为零，导致下列问题：

$$\rho C \frac{\partial u}{\partial t} - \nabla \cdot (k \nabla u) = f$$

其中，ρ为密度，C 为棒的热能，k 为热传导率，f 为放射性热源。

棒的密度为 7 800kg/m^3，热能为 500Ws/kg • ℃，热传导率为 40W/m • ℃，热源为 20000W/m^3。右端温度为 100℃，外边界周边温度为 100℃，热交换系数（表面传热系数）为 50W/m^2 • ℃，左端的热流量为 5000W/m^2。

因为这是一个圆柱问题，所以需要将方程改为由 r，z 和θ所确定的柱坐标的形式。由于对称性，解与θ无关，所以方程可转换为下面的形式：

$$r\rho C \frac{\partial u}{\partial t} - \frac{\partial}{\partial r}(kr\frac{\partial u}{\partial r}) - \frac{\partial}{\partial z}(kr\frac{\partial u}{\partial z}) = fr$$

边界条件为：

（1）$\bar{n} \cdot (k\nabla u) = 5000$（左端，Neumann 条件）

因为广义 Neumann 条件为$\bar{n} \cdot (C\nabla u) + qu = g$，而且本问题中 C 取决于 r（C=kr），所以边界条件可表达为：$\bar{n} \cdot (C\nabla u) = 5000r$ 。

（2）u=100（右端，Dirichlet 条件）

（3）$\bar{n} \cdot (k\nabla u) = 50(100 - u)$（外边界，广义 Neumann 条件），必须表达为

$$\bar{n} \cdot (C\nabla u) + 50ru = 50r \cdot 100$$

（4）在原问题中，柱轴 r =0 不是边界，但现在降维成二维问题以后，它是边界，所以必

须给出人为边界条件：$\bar{n}\cdot(C\nabla u)=0$。

初值为 $u(t0)$=0。

本问题的解如图 28-10 所示。

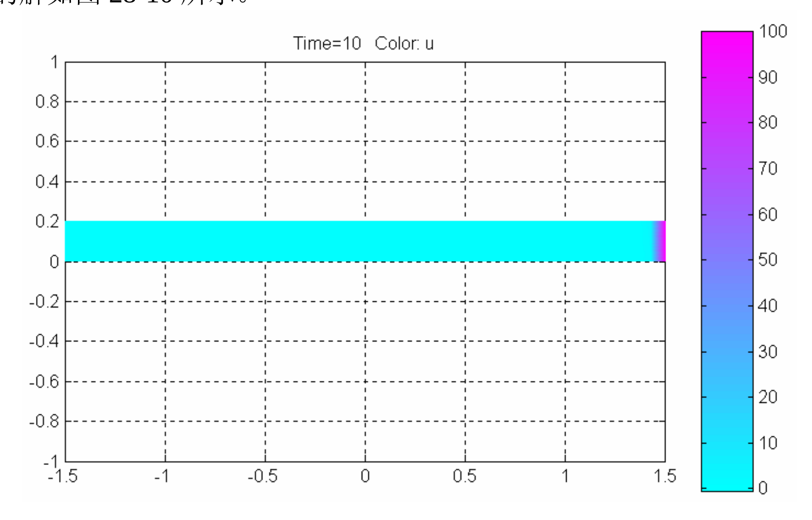


图 28-10　放射性棒热扩散问题的解

使用 GUI 按照下面步骤求解。

打开 PDE 图形用户界面，建立几何模型。

将棒体画成一个基线平行于 X 轴的矩形，并令 x 轴为 z 方向，Y 轴为 r 方向。矩形四角的坐标分别为(-1.5, 0),(1.5, 0),(1.5, 0.2),(-1.5, 0.2)，则棒的长度为 3，半径为 0.2。

双击边界，打开 Boundary Condition 对话框，在对话框中输入边界条件。对于左边界，输入 Neumann 条件：q=0, g=5000*y。右端用 Dirichlet 条件：h=1, r=100。对于外边界，用 Neumann 条件：q=50*y,g=50*y*100，坐标轴用 Neumann 条件：q=0, g=0。在 PDE Specification 对话框中输入系数：c=40*y,a=0,d=7800*500*y,f=20000*y。

在 20000 s 范围内加亮解图形（每 1000 s 计算一次），可以观察热量在右侧和外边界上是怎样传播的。

打开 PDE Specification 对话框，将 PDE 类型改为 Elliptic，则显示 u 与时间无关时的解，即稳态解。如图 28-11 所示。

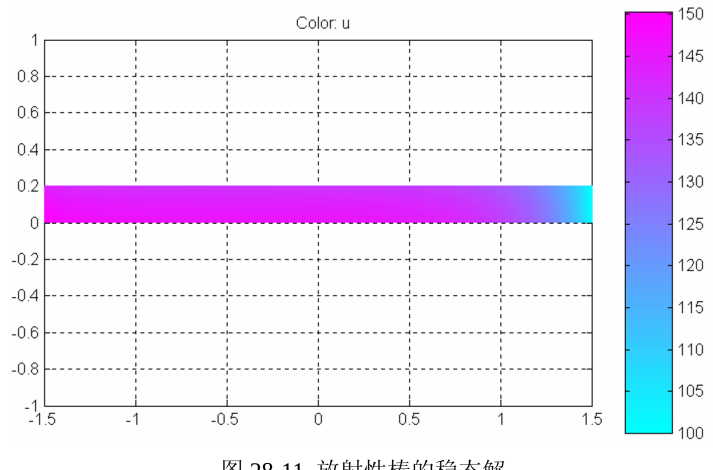


图 28-11 放射性棒的稳态解

设置热交换系数为零，可以演示外边界的冷却效果，如图 28-12 所示。

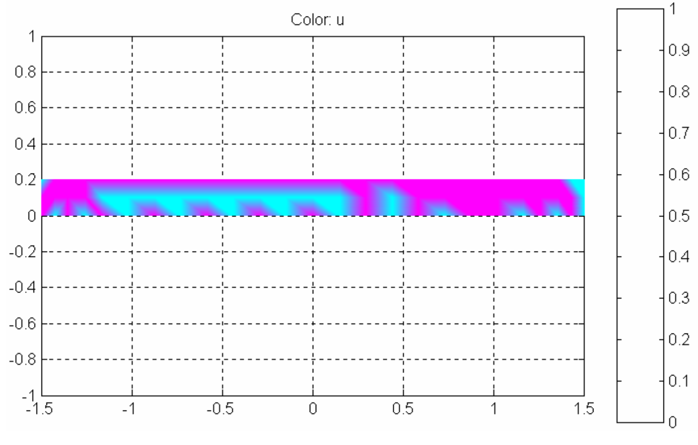


图 28-12　外边界的冷却效果

28.3　双曲线型问题

28.3.1　波动方程

四角坐标为(-1,-1),(-1,1),(1,-1)和(1,1)的方形上薄膜的横向振动的波动方程可表达为

$$\frac{\partial^2 u}{\partial t^2} - \Delta u = 0$$

薄膜左侧和右侧固定（u=0），上端和下端自由($\frac{\partial u}{\partial n} = 0$)。另外，需要知道初值 $u(t_0)$和 $\frac{\partial u(t_0)}{\partial t}$。若 t=0，则 $u(0) = \arctan(\cos(\frac{\pi}{2}x))$ 和 $\frac{\partial u(0)}{\partial t} = 3\sin(\pi x)\mathrm{e}^{\sin(\frac{\pi}{2}y)}$ 为满足边界条件的初值。

28.3.2　波动方程的求解

1．用 GUI

用 GUI 按照下面的步骤进行求解：

① 画矩形：在 Draw 菜单中单击 Rectangle/square 选项。

② 确定边界条件：单击∂Ω按钮或双击边界定义边界条件。

③ 网格初始化：单击△按钮或 Mesh 菜单中的 Initialize Mesh 选项。

④ 定义双曲线型 PDE：单击PDE按钮，打开 PDE Specification 对话框，选择 Hyperbolic PDE，输入合适的系数值，一般的双曲线型 PDE 描述为

$$d\frac{\partial^2 u}{\partial t^2} - \nabla \cdot (c\nabla u) + au = f$$

所以 c=1, a=0, f=0, d=1。

⑤进行求解：在 Solve 菜单中选择 Parameters…选项，打开 Solve Parameters 对话框，输入 linspace(0,5,31)，作为时间列表，输入 u 的初值：

atan(cos(pi/2*x))

以及 $\frac{\partial u}{\partial t}$ 的初值：

3*sin(pi*x).*exp(sin(pi/2*y))

最后单击 = 按钮，进行求解。

通过照亮整个解序列可以观察波在 *x* 和 *y* 方向上的传播。

2. 用命令行函数

几何模型-squareg.m，边界条件-squareb3.m。用下面的命令行进行求解。首先，创建网格并定义初值和时间。

```
[p, e, t]=initmesh('squareg');

x=p(1,:)';
y=p(2,:)';

u0=atan(cos(pi/2*x));
ut0=3*sin(pi*x).*exp(sin(pi/2*y));

n=31;
tlist=linspace(0, 5, n);    %时间列表
```

下面设置求解参数：

```
uu=hyperbolic(u0, ut0, tlist, 'squareb3', p, e, t, 1, 0, 0, 1);
Time: 0.166667
Time: 0.333333
Time: 0.5
Time: 0.666667
Time: 0.833333
Time: 1
Time: 1.16667
Time: 1.33333
Time: 1.5
Time: 1.66667
Time: 1.83333
Time: 2
Time: 2.16667
Time: 2.33333
Time: 2.5
Time: 2.66667
Time: 2.83333
Time: 3
Time: 3.16667
Time: 3.33333
Time: 3.5
Time: 3.66667
Time: 3.83333
Time: 4
Time: 4.16667
Time: 4.33333
Time: 4.5
Time: 4.66667
Time: 4.83333
```

```
Time: 5
428 successful steps
62 failed attempts
982 function evaluations
1 partial derivatives
142 LU decompositions
981 solutions of linear systems
```

然后使解可视化。加入矩形网格可以加速绘图。

```
delta=-1:0.1:1;
[uxy, tn, a2, a3]=tri2grid(p,t,uu(:, 1), delta, delta);
gp=[tn; a2; a3];

umax=max(max(uu));
umin=min(min(uu));

newplot
M=moviein(n);
for i=1:n
  pdeplot(p, e, t, 'xydata', uu(:,1), 'zdata', uu(:,1),…
          'mesh', 'off', 'xygrid', 'on', 'gridparam', gp,…
          'colorbar', 'off', 'zstyle', 'continuous');
  axis([-1 1 –1 1 umin umax]);
  caxis([umin umax]);
  M(:, i)=getframe;
end
movie(M, 10);
```

结果如图 28-13 所示。

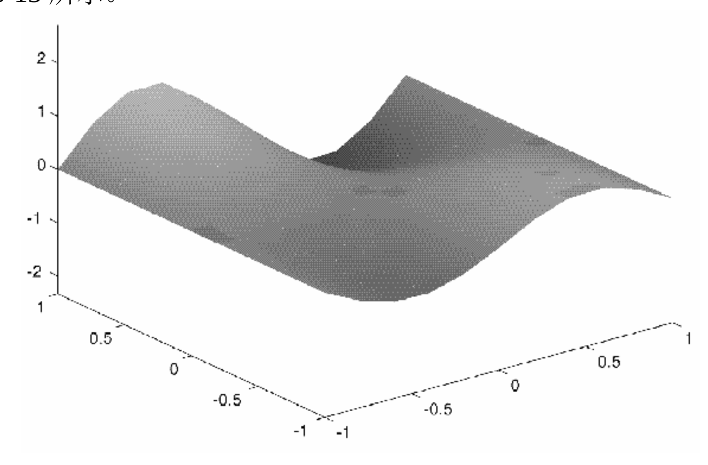


图 28-13　波动方程的解

28.4　特征值问题

28.4.1　L 形薄膜的特征值和特征函数

对于 L 形薄膜的特征值模式 PDE 问题：

$$-\Delta u = \lambda u$$

计算所有特征值<100m 的特征模式，边界上 u=0(Dirichlet 条件)。

1．使用 GUI

打开 PDE 的 GUI 以后，选择 Generic Scalar 模式，用□按钮画 L 形薄膜，角点位置为(0, 0),(-1, 0), (-1, -1), (1, -1), (1, 1)和(0, 1)。

本问题的边界条件按默认设置。

初始化网格，细化两次。

打开 PDE Specification 对话框，选择特征模式，PDE 系数的默认值 c=1, a=0, d=1 均与本问题的要求一致，故直接单击 OK 按钮退出即可。

在 Solve 菜单条中选择 Parameters…选项，打开 Solve Parameters 对话框。对话框中有一个编辑框，用于输入特征值搜索范围。默认输入为[0 100]，接受默认值。

最后，单击=按钮，求解本问题。

解显示的是第 1 个特征函数，第 1 个特征值（最小值）也显示出来。在 GUI 底部的信息栏中可以看到特征值的个数。可以打开 Plot Selection 对话框，从弹出式菜单中选择对应特征值来决定绘哪一个特征函数的图形。

2．用命令行函数

几何模型的 M 文件为 lshapeg.m。边界条件的 M 文件为 lshapeg.m。

首先，初始化网格，改进两次。

```
[p, e, t]=initmesh('lshapeg');
[p, e, t]=refinemesh('lshapeg', p, e, t);
[p, e, t]=refinemesh('lshapeg', p, e, t);
```

通过调用 pdeeig 函数来找出特征值的个数。

```
[v, l]=pdeeig('lshapeb', p, e, t, 1, 0, 1, [0 100]);
                 Basis= 10,  Time=   1.92,  New conv eig=  0
                 Basis= 13,  Time=   2.36,  New conv eig=  0
                 Basis= 16,  Time=   2.85,  New conv eig=  0
                 Basis= 19,  Time=   3.40,  New conv eig=  1
                 Basis= 22,  Time=   4.00,  New conv eig=  2
                 Basis= 25,  Time=   4.66,  New conv eig=  3
                 Basis= 28,  Time=   5.38,  New conv eig=  3
                 Basis= 31,  Time=   6.15,  New conv eig=  5
                 Basis= 34,  Time=   6.97,  New conv eig=  5
                 Basis= 37,  Time=   7.85,  New conv eig=  7
                 Basis= 40,  Time=   8.95,  New conv eig=  7
                 Basis= 43,  Time=   9.94,  New conv eig= 10
                 Basis= 46,  Time=  11.04,  New conv eig= 11
                 Basis= 49,  Time=  12.19,  New conv eig= 11
                 Basis= 52,  Time=  13.45,  New conv eig= 13
                 Basis= 55,  Time=  14.77,  New conv eig= 14
                 Basis= 58,  Time=  16.36,  New conv eig= 14
                 Basis= 61,  Time=  17.79,  New conv eig= 15
                 Basis= 64,  Time=  19.38,  New conv eig= 16
                 Basis= 67,  Time=  21.03,  New conv eig= 18
```

```
                Basis= 70,  Time=  22.73,  New conv eig= 21
End of sweep:  Basis= 70,  Time=  22.73,  New conv eig= 21
                Basis= 31,  Time=  25.43,  New conv eig=  0
                Basis= 34,  Time=  26.25,  New conv eig=  0
                Basis= 37,  Time=  27.07,  New conv eig=  0
End of sweep:  Basis= 37,  Time=  27.13,  New conv eig=  0
```

有 19 个特征值<100。绘出第 1 个特征值模式，将它与 MATLAB 的 membrane 函数比较。

```
pdesurf(p, t, v(:, 1))
```

结果如图 28-14 所示。

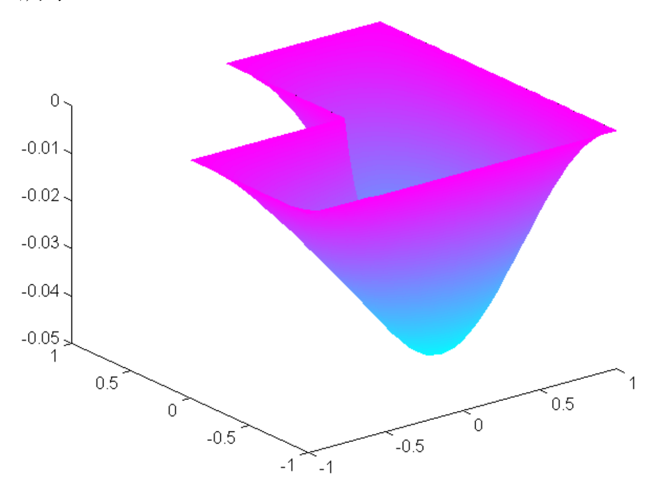


图 28-14　第 1 个特征值模式的图形

```
figure
membrane(1, 20, 9, 9)
```

结果如图 28-15 所示。

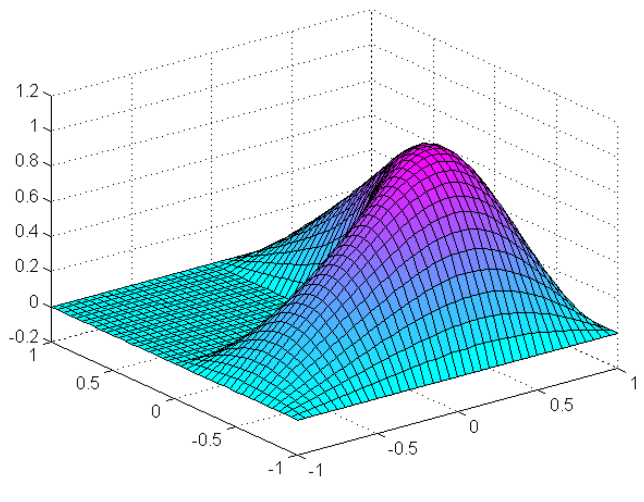


图 28-15　L 形薄膜图

membrane 函数可以为 L 形薄膜生成前 12 个特征函数，还可以比较第 12 个特征值模式。

```
figure
pdesurf(p, t, v(:, 12))
```

结果如图 28-16 所示。

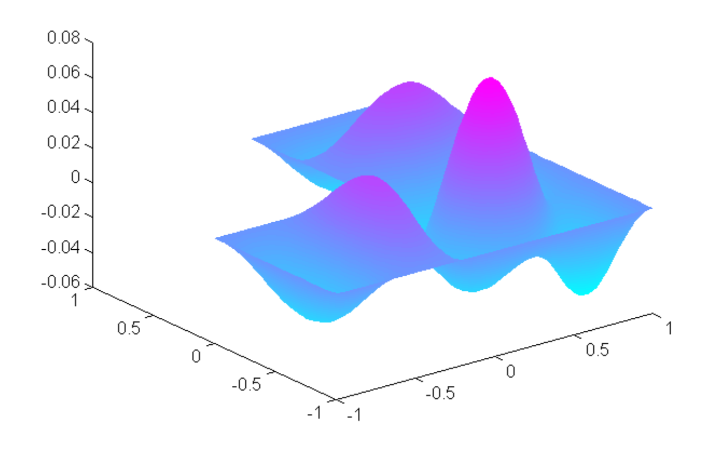


图 28-16　第 12 个特征值模式

```
figure
membrane(12, 20, 9, 9)
```

结果如图 28-17 所示。

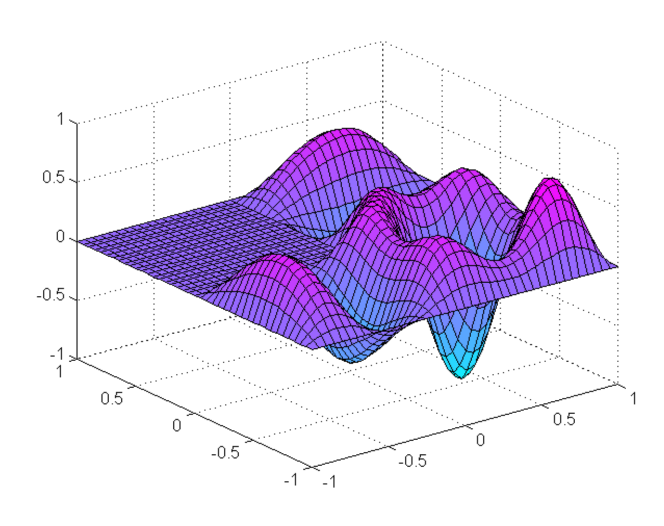


图 28-17　第 12 个特征值模式的薄膜图

28.4.2　圆角 L 形薄膜

通过在 L 形薄膜的角部添加圆弧，可以获得圆角效果。几何模型函数 lshapec.m 为

pdepoly([-1, 1, 1, 0, 0, -1], [-1, -1, 1, 1, 0, 0], 'p1');

pdecirc(-a, a, a, ’C1’);

pderect([-a 0 a 0], ’SQ1’);

后面两个函数与输入变量半径 a 有关，其默认值为 0.5，可以先指定。

当半径 a 增大时，特征值和对应特征值模式的频数减小，L 形薄膜的形状变得更圆。

图 28-18 中显示了圆角 L 形薄膜的第 1 个特征值模式。

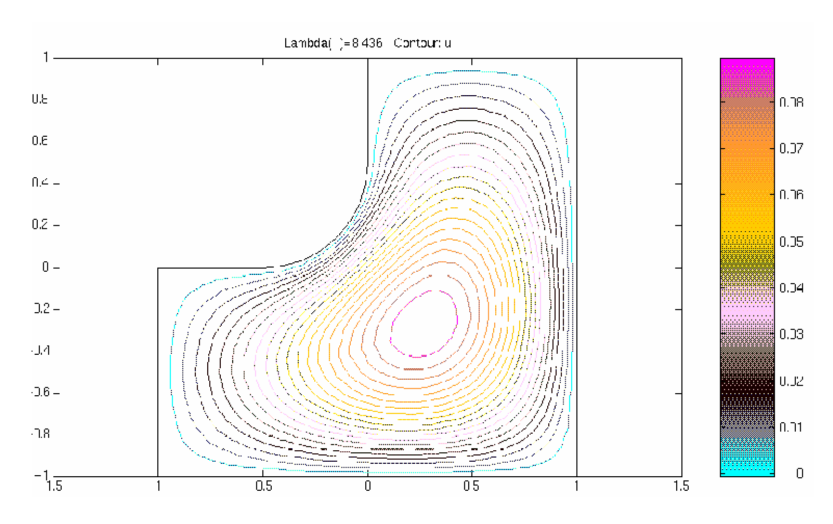


图 28-18　圆角 L 形薄膜的第 1 个特征值模式解的等值线图

28.4.3　方形的特征值和特征值模式

方形的四角为(-1,-1), (-1,1), (1,1)和(1,-1)。边界条件为

（1）左边界　u=0(Dirichlet 条件)

（2）上下边界　$\frac{\partial u}{\partial n}=0$　（Neumann 条件）

（3）右边界　$\frac{\partial u}{\partial n}-\frac{3}{4}u=0$　(Neumann 条件)

特征值 PDE 问题为

$$-\Delta u=\lambda u$$

我们希望找到小于 10 的特征值及其对应的特征模式，所以搜索范围为[-Inf 10]。

1．使用 GUI

首先进入图形用户界面，绘几何模型。Draw-Rectangle/square 或单击□按钮，画方形。

然后定义边界条件　单击∂Ω按钮，双击边界，定义边界条件。在右边界输入 Neumann 条件：g=0, q=-34。

将网格初始化并改进它。

定义特征值 PDE 问题：打开 PDE Specification 对话框，选择 Eigenmodes 单选钮。

求解参数取默认值：c=1, a=0, d=1。

在 Solve Parameters 对话框中输入特征值的范围[-Inf 10]。

最后，单击=按钮，进行求解。默认时，绘出第 1 个特征函数的图形。

2．使用命令行函数

几何描述的 M 文件为 squareg.m。

边界条件的 M 文件为 squareb2.m。

用下面的命令找到指定范围内的特征值和对应的特征函数。

```
[p, e ,t]=initmesh('squareg');
[p, e, t]=refinemesh('squareg', p, e, t);
```

特征值 PDE 系数为：c=1, a=0, d=1。

特征值范围为[-Inf, 10]。

pdeeig 函数返回两个输出变量，数组 l 为特征值，矩阵 V 中为对应的特征函数。

```
[v, l]=pdeeig('squareb2', p, e, t, 1, 0, 1,[-Inf 10]);
                  Basis= 10,  Time=  1.65,  New conv eig=  0
                  Basis= 17,  Time=  2.31,  New conv eig=  2
                  Basis= 24,  Time=  3.02,  New conv eig=  8
End of sweep:  Basis= 24,  Time=  3.08,  New conv eig=  8
                  Basis= 18,  Time=  3.96,  New conv eig=  0
End of sweep:  Basis= 18,  Time=  4.01,  New conv eig=  0
```

绘制第 4 个特征函数，输入

```
pdesurf(p, t, v(:, 4))
```

结果如图 28-19 所示。

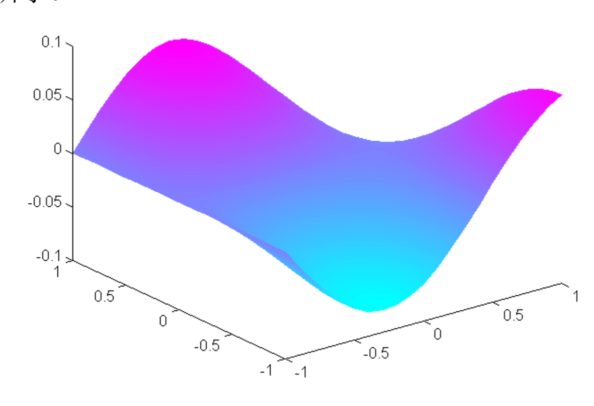


图 28-19　第 4 个特征函数的色谱图

第 29 章　应 用 模 式

29.1　概述

MATLAB 的偏微分方程数值解工具箱可以应用于结构力学、静电学、静磁学、电磁学、热传导和扩散问题等许多领域。当应用于不同的领域时会有不同的方程和参数。为了方便应用，图形用户界面中提供了应用模式的下拉式列表框，可以根据实际问题在其中进行选择。可用的应用模式有：

（1）Generic scalar（默认选项）—— 一般标量问题；

（2）Generic System—— 一般系统问题；

（3）Structural Mechanics-Plane Stress——结构力学−平面应力问题；

（4）Structural Mechanics-Plane Strain——结构力学−平面应变问题；

（5）Electrostatics——静电学；

（6）Magnetostatics——静磁学；

（7）AC Power Electromagnetics——交流电电磁学；

（8）Conductive Media DC——直流导电介质；

（9）Heat Transfer——热传导；

（10）Diffusion——扩散问题。

第 1 类和第 2 类问题在前面已经进行了介绍，下面分别介绍后面 8 种应用模式。

29.2　结构力学——平面应力

在结构力学中，应力—应变之间的关系可以表达为

$$\begin{bmatrix}\sigma_x\\ \sigma_y\\ \tau_{xy}\end{bmatrix}=\frac{E}{1-v^2}\begin{bmatrix}1 & v & 0\\ v & 1 & 0\\ 0 & 0 & (1-v)/2\end{bmatrix}\begin{bmatrix}\varepsilon_x\\ \varepsilon_y\\ \gamma_{xy}\end{bmatrix}$$

其中，σ_x 和 σ_y 分别是 x, y 方向上的应力。τ_{xy} 为剪应力，E 为杨氏模量，v 为泊松比。材料的变形用 x, y 方向的位移 u 和 v 描述。因此应变可定义为

$$\varepsilon_x=\frac{\partial u}{\partial x},\ \varepsilon_y=\frac{\partial v}{\partial y},\ \gamma_{xy}=\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}$$

力的平衡方程为

$$-\frac{\partial\sigma_x}{\partial x}-\frac{\partial\tau_{xy}}{\partial y}=k_x$$
$$-\frac{\partial\tau_{xy}}{\partial x}-\frac{\partial\sigma_y}{\partial y}=k_y$$

式中，k_x和k_y为体力。

组合以上关系，可以获得变形方程，记为

$$-\nabla \cdot (\underline{c} \otimes \nabla u) = \underline{k}$$

式中$\underline{c}$可以写成4个2×2的矩阵$\boldsymbol{c}_{11}$，$\boldsymbol{c}_{12}$，$\boldsymbol{c}_{21}$和$\boldsymbol{c}_{22}$。

$$\boldsymbol{c}_{11} = \begin{pmatrix} 2G+\mu & 0 \\ 0 & G \end{pmatrix}$$

$$\boldsymbol{c}_{12} = \begin{pmatrix} 0 & \mu \\ G & 0 \end{pmatrix}$$

$$\boldsymbol{c}_{21} = \begin{pmatrix} 0 & G \\ \mu & 0 \end{pmatrix}$$

$$\boldsymbol{c}_{22} = \begin{pmatrix} G & 0 \\ 0 & 2G+\mu \end{pmatrix}$$

式中，G为剪切模量，定义为

$$G = \frac{E}{2(1+v)}$$

于是可以定义

$$\mu = 2G\frac{v}{1-v}$$

$$\underline{k} = \begin{bmatrix} k_x \\ k_y \end{bmatrix}$$

为体力。

这是一个椭圆型PDE问题，下一步需要做的是选择应用模式Structural Mechanics-Plane Stress，并在PDE Specification对话框中输入材料参数E和v以及体力。

在本模式中，也可以求解特征值问题，问题描述为

$$-\nabla \cdot (\underline{c} \otimes \nabla u) = \lambda \underline{d} u$$

$$\underline{d} = \begin{pmatrix} \rho & 0 \\ 0 & \rho \end{pmatrix}$$

ρ为密度，也可以在PDE Specification对话框中输入。

在Plot Selection对话框中进行选择，x和y方向的位移u和v，以及向量（u,v）的绝对值可以通过颜色、等值线或z向高度来表示，位移向量场(u,v)可以用箭头或变形网格表示。下面15种量可以用颜色、等高线或高度来表示：

（1）$ux = \dfrac{\partial u}{\partial x}$

（2）$uy = \dfrac{\partial u}{\partial y}$

（3）$vx = \dfrac{\partial v}{\partial x}$

（4）$vy = \dfrac{\partial v}{\partial y}$

（5）exx　　x 方向的应变（ε_x）

（6）eyy　　y 方向的应变（ε_y）

（7）exy　　剪应变(γ_{xy})

（8）S_{xx}　　x 方向的应力（σ_x）

（9）S_{yy}　　y 方向的应力（σ_y）

（10）S_{xy}　　剪应力（τ_{xy}）

（11）e_1　　第 1 主应变（ε_1）

（12）e_2　　第 2 主应变（ε_2）

（13）S_1　　第 1 主应力（σ_1）

（14）S_2　　第 2 主应力（σ_2）

（15）von Mises　　冯·米色斯有效应力（$\sqrt{\sigma_1^2+\sigma_2^2-\sigma_1\sigma_2}$）

【例 29-1】　一块钢板，其左下角被夹住，在右上角沿圆弧形裂缝拉伸，其他各边自由。钢板具有以下性质：大小为 1m×1m，厚 1mm；被夹住的部分为 1/3m×1/3m，圆弧形裂缝从（2/3,1）延伸至（1，2/3），杨氏模量为 196×10^3MN/m^2，泊松比为 0.31，曲线边界承受向外的正压力 500N/m，除以厚度 1mm，即得表面拉应力 0.5MN/m^2。求 x 方向和 y 方向的应变和应力、剪应力和冯·米色斯有效应力。

使用 GUI，按照下面步骤进行求解。

（1）打开图形用户界面，选择 Structural Mechanics, Plane Stress 应用模式。然后建模。建模的第 1 步是绘多边形，其各角点的坐标分别为 x=[0 2/3 1 1 1/3 1/3 0]和 y=[1 1 2/3 0 0 1/3 2/3]。第 2 步是以（2/3 2/3）为圆心，以 1/3 为半径画圆。多边形的标签为 P1，圆的标签为 C1，P1+C1 则为钢板的 CSG 模型。

（2）选择 Boundary Mode，指定边界条件。

首先在 Boundary 菜单中选择 Remove All Subdomain Borders…选项，删除所有的次级域边界。左下方插入部分的两个边界的边界条件为 Dirichlet 条件，位移为零。圆形裂纹处为 Neumann 条件，q=0, g1=0.5*nx, g2=0.5*ny。其余边界为自由边界（无正压力），即为 Neumann 条件：q=0, g=0。

（3）打开 PDE Specification 对话框，输入 PDE 参数。

E 和 v 分别为杨氏模量和泊松比。无体力，故 k_x 和 k_y 为零。ρ 在此模式中没有应用。材料为均质，所以相同的 E 和 v 适用于整个二维分析域。

单击按钮，初始化网格。

单击按钮，精细化网格。

单击按钮，进行求解。

可以针对很多不同的应变和应力属性进行图形输出。例如位移 u,v，x 方向和 y 方向的应变和应力、剪应力、冯·米色斯有效应力和主应力、主应变等。实现图形化的方法是在 Plot Selection 对话框中的弹出式菜单中选择这些属性。也可以通过选择 Color，Weight，vector field arrows 和 displacements in a 3-D plot 来同时显示不同数量与向量的组合。

选择 Color 核选框和 Deformed Mesh 图形，生成颜色区分的冯·米色斯有效应力图和用

变形网格表示的位移向量场图。在 Color 行的弹出式菜单中选择 Von Mises 选项，绘制冯·米色斯有效应力图。

在解（应力）的梯度比较大的地方，需要细化网格，以提高解的精度。在 Solve 菜单中选择 Parameters…选项，并选择 Adaptive Mode 单选钮。

选择默认选项 Worst triangles 三角形选择方法，将 Worst triangle fraction 设为 0.5。现在重新求解平面应力问题。在 Plot Selection 对话框中选择 Show Mesh 单选钮，看应力大的地方网格是如何细化的。

自适应细化网格如图 29-1 所示。

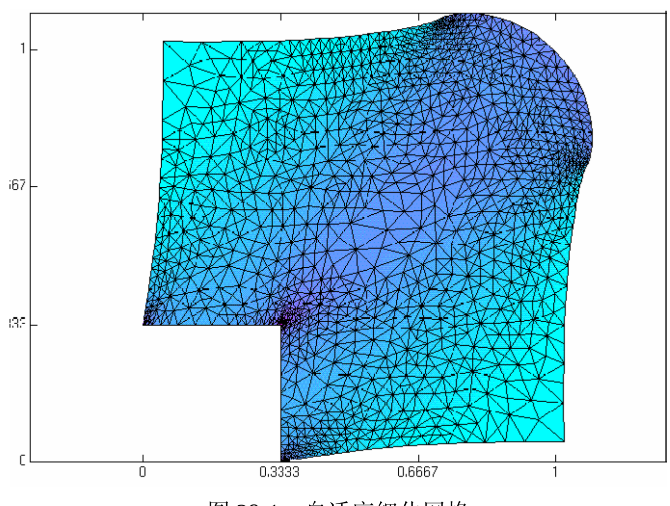


图 29-1　自适应细化网格

29.3　结构力学——平面应变

当 Z 轴的位移为零，x 和 y 方向的变形仅为 x 和 y 的函数时的变形状态称为平面应变。可以通过选择 Structural Mechanics，Plane Strain 应用模式来利用 PDE 工具箱解决平面应变问题。

本模式中，应力-应变的关系与平面应力模式相比只有细微的区别，因而可以沿用相同的参数。这两个模式具有相同的界面。

本模式中与平面应力模式中不同的方程有：

（1）$\underline{c}$ 中的μ 参数由下式定义：

$$\mu = 2G\frac{v}{1-2v}$$

（2）冯·米色斯有效应力用下式计算：

$$\sqrt{(\sigma_1^2+\sigma_2^2)(v^2-v+1)-\sigma_1\sigma_2(2v^2-2v-1)}$$

平面应变问题不如平面应力问题普遍。一个典型的例子是沿 Z 轴展布的地下隧硐的某个横断面。它的变形基本上符合平面应变条件。

29.4　静电学

与静电学相关的应用设施包括高电压设施、电子设备和电容器等。“Statics”表示时间的

变化率很小，与分析域的大小相比，波长很大。在电静力学中，电势 V 与电场强度 E 之间的关系为 $E=-\nabla V$ ，利用麦克斯韦尔方程 $\nabla \cdot D=\rho$ 和关系式 $D=\varepsilon E$ ，得到泊松方程

$$-\nabla \cdot (\varepsilon \nabla V)=\rho$$

式中，ε 为介电系数，ρ 为空间电荷密度。

进入图形用户界面，选择 Electrostatics 应用模式，可以求解用上面方程描述的静电学问题。

PDE Specification 对话框包含有 ε 和 ρ 的输入。

电静力学的边界条件可以是 Dirichlet 条件或 Neumann 条件。若为 Dirichlet 条件，则电势 V 被指定在边界上。对于 Neumann 条件，表面电荷 $n \cdot (\varepsilon \nabla V)$被指定在边界上。

图形化输出的选项有电势 V、电场强度 E 和电位移场 D 等。

【例 29-2】 一个内部充满空气的方框，内边长为 0.2m，外边长为 0.5m。内边界上电势为 1000 V，外边界上电势为 0 V，域内没有电荷。现求方框中的电势。

求解此问题，即求解拉普拉斯方程：

$$\nabla V=0$$

边界条件为 Dirichlet 条件，内边界上 V=1000，外边界上 V=0。

下面用 GUI 求解比问题。

在命令窗口输入 pdetool 命令，打开图形用户界面，选择 Electrostatics 模式并建立几何模型。

首先画一个边长为 0.2 的方形（用 Snap 单选钮并调整网格间距），然后画一个中心位置相同的边长为 0.5 的方形。若第一个方形名为 SQ1，第二个方形名为 SQ2，则分析域为 SQ2-SQ1。

在 Set formula 编辑框中输入表达式，然后输入边界条件。用 Shift-单击来选择所有的内边界，然后双击内边界并输入 1000 作为 Dirichlet 边界。

然后打开 PDE Specification 对话框，并在 Space charge density 编辑域中输入 0。介电系数（Dielectricity）设为 1。

将网格初始化，单击△按钮求解方程。用 adaptive 模式，可以改进解的精度。

例如，选择三角形选择方法为 Worst triangles。设置三角形的最大数目为 500。单击 Refine 按钮来细化网格。最后将 adaptive 模式设置为 off，多次单击=按钮。

在 Plot Selection 对话框中选择 a contour plot，可以查看等势线；在 Contour plot levels 编辑框中输入 0:100:1000，则每隔 100 显示一条等势线。如图 29-2 所示。

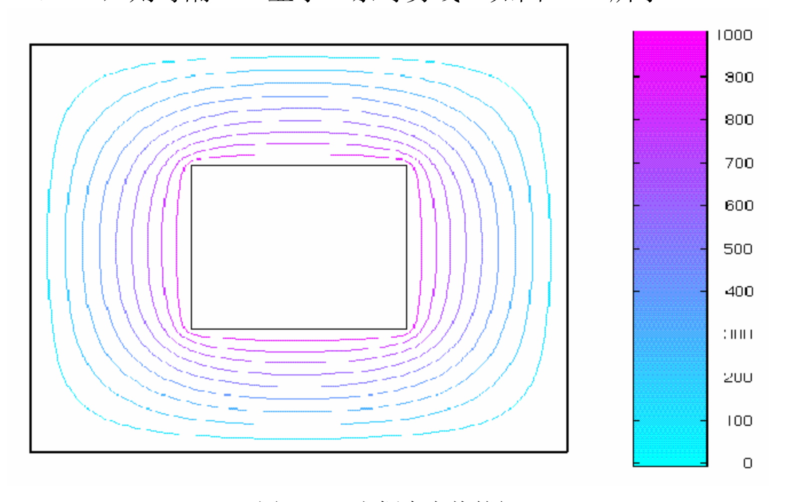


图 29-2　方框中电势的解

29.5　静磁学

磁铁、电动机和运输工具等都存在静磁学问题。“Statics”表示时间变化率较小，因此对于稳定情况，我们从麦克斯韦尔方程开始。

$$\nabla \times \boldsymbol{H} = \boldsymbol{J}$$

$$\nabla \cdot B = 0$$

B 和 **H** 之间存在下面的关系：

$$\boldsymbol{B} = \mu \boldsymbol{H}$$

式中，**B** 为磁感应强度，**H** 为磁场强度，**J** 为磁化强度。μ 为材料的磁导率。

因为$\nabla \cdot B = 0$，所以存在一个磁矢势 **A**，使得

$$\boldsymbol{B} = \nabla \times \boldsymbol{A}$$

且

$$\nabla \times (\frac{1}{\mu} \nabla \times \boldsymbol{A}) = \boldsymbol{J}$$

平面问题假设电流与 Z 轴平行，所以只有 **A** 的 z 分量存在。

$$\boldsymbol{A}=(0, 0, A)，\boldsymbol{J}=(0, 0, J)$$

上面的公式可以简化为椭圆型 PDE 问题：

$$-\nabla \cdot (\frac{1}{\mu} \nabla \boldsymbol{A}) = \boldsymbol{J}$$

式中，$\boldsymbol{J}=J(x, y)$

对于二维问题，可以计算磁感应强度 **B**。

$$\boldsymbol{B}=\left[\frac{\partial A}{\partial y}, \frac{\partial A}{\partial x}, 0\right]$$

和磁场强度 **H**

$$\boldsymbol{H} = \frac{1}{\mu} \boldsymbol{B}$$

【例 29-3】　考虑二极电动机中由于定子（stator）转动引起的静磁场。电动机较长，忽略端部效应，因此二维计算模型已经够了。

分析域包括 4 部分：两个铁磁片（定子和转子）、两者之间的气隙和携带直流电的电枢线圈。

在空气和线圈中，磁导率为 1，在定子和转子中，磁导率由下式定义：

$$\mu = \frac{\mu_{\max}}{1 + c|\nabla(A)|^2} + \mu_{\min}$$

$\mu_{\max} = 500, \mu_{\min} = 200, c = 0.05$时，可代表矽钢片。

除了线圈中，其他任何地方的电流密度 J 为 0，线圈中为 1。

由于问题的对称性，可以提取分析域的第 1 象限（$x \geqslant 0, y \geqslant 0$）进行分析，X 轴上为 Neumann 条件：$n \cdot \left[\frac{1}{\mu} \nabla A\right] = 0$，Y 轴上为 Dirichlet 条件：$A$=0。电动机以外的区域被忽略，所以外部

边界上为 Dirichlet 边界条件，即 A=0。

下面用 GUI 进行求解。

该问题的几何模型较复杂，由 5 个圆和两个矩形组成。用 pdetool GUI，将 x 轴和 y 轴分别限于[-1.5 1.5]和[-1 1]中。设置应用模式为 Magnetostatics，网格间隔为 0.1。使用“Snap-to-grid”特色，可以用鼠标画出几何图形，也可以通过输入下面的命令来画图：

```
pdecirc(0, 0, 1, 'C1')
pdecirc(0, 0, 0.8, 'C2')
pdecirc(0, 0, 0.6, 'C3')
pdecirc(0, 0, 0.5, 'C4')
pdecirc(0, 0, 0.4, 'C5')
pderect([-0.2 0.2 0.2 0.9],'R1')
pderect([-0.1 0.1 0.2 0.9],'R2')
pderect([0 1 0 1],'SQ1')
```

可以获得一个如图 29-3 所示的模型。

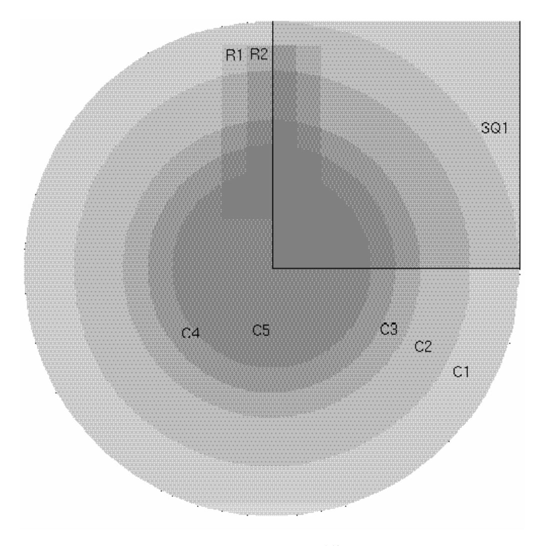


图 29-3　几何模型

输入下式来获取该模型的四分之一圆周：

```
(C1+C2+C3+C4+R1+R2)*SQ1
```

在边界模式中，需要删除一些次级区域的边界，用 shift-单击选择边界，并用 Boundary 菜单中的 Remove Subdomain Border 选项删除它们，直到几何模型由四部分组成，即定子、转子、线圈和气隙域。图 29-4 中，定子为次级区域 1，转子为次级区域 2，线圈为次级区域 3，气隙为次级区域 4。

在移动到 PDE 模式以前，选择沿 X 轴的边界并设置边界条件 Neumann 条件，即 g=0, q=0。在 PDE 模式中，在 PDE 菜单中选择 Show Subdomain Labels 选项来设置标签。在每一个次级区域中双击，定义 PDE 参数。

（1）在线圈中，μ 和 J 都为 1，用默认值即可。

（2）在定子和转子中，μ为非线性的，由前面求磁导率的等式定义，用下面的形式输入μ：

$$500./(1+0.05*(ux.\hat{}2+uy.\hat{}2))+200ux.\hat{}2+uy.\hat{}2=|\nabla A|^2$$

J 为 0，表示没有电流。

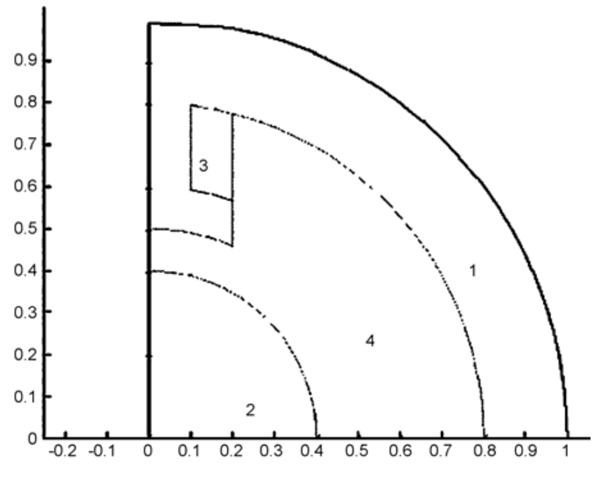


图 29-4　边界条件设置

（3）在气隙域中，μ=1，J=0。

将网格初始化，打开 Solve Parameters 对话框。在 Solve 菜单中选择 Parameters 选项。因为存在非线性问题，所以必须选择 Use nonlinear solver 核选框来启用非线性求解器。

可以调整容限参数，adaptive 求解器可与非线性求解器一起使用，求解 PDE 并用箭头绘制磁感应强度 $\boldsymbol{B}$，用等值线图表示静磁势的等势线 A，如图 29-5 所示。图中清楚地表示出了磁感应强度与静磁势等势线之间的平行关系。

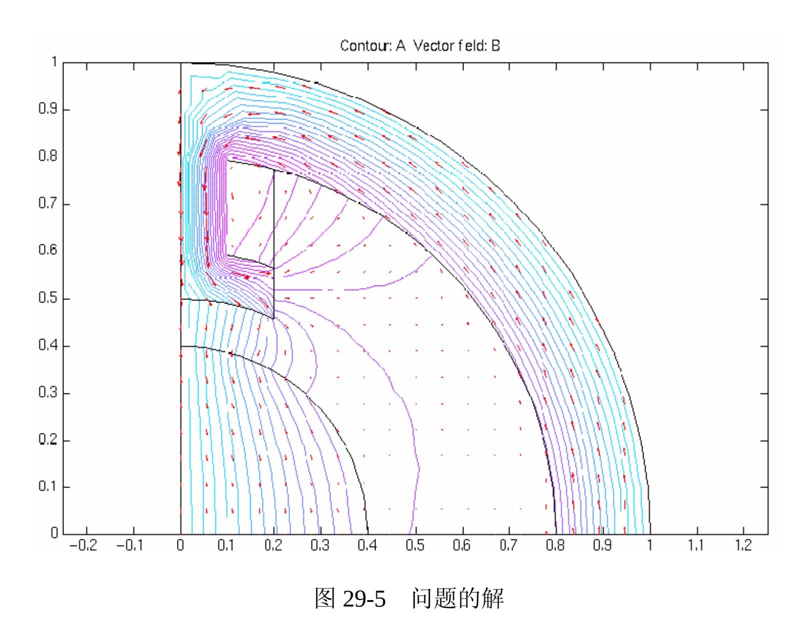


图 29-5　问题的解

29.6　交流电电磁学

交流电电磁学问题是在研究电动机、运输工具和带交流电的导体时发现的。

首先，考虑一个均匀介质，其介电系数为ε，磁导率为μ，在任意点上无电荷，该区域必须满足一般麦克斯韦尔方程的一个特殊组合：

$$\nabla \times E = -\mu \frac{\partial H}{\partial t}$$

$$\nabla \times H = \varepsilon \frac{\partial E}{\partial t} + J$$

没有电流时，可以从第 1 个集合中剔除 H，从第 2 个集合中剔除 E，并看到两个区域都满足波速为$1/\sqrt{\varepsilon\mu}$ 的波动方程。

$$\Delta E - \varepsilon\mu \frac{\partial^2 E}{\partial t^2} = 0,\ \Delta H - \varepsilon\mu \frac{\partial^2 H}{\partial t^2} = 0$$

然后研究一个电荷自由分布的均质介质，电导率为σ，则电流密度为

$$J = \sigma E$$

且波被欧姆电阻阻尼：

$$\Delta E - \mu\sigma \frac{\partial E}{\partial t} - \varepsilon\mu \frac{\partial^2 E}{\partial t^2} = 0$$

$\boldsymbol{H}$ 的情况与此近似。

对于时间谐和场， $\boldsymbol{E} = E_c \mathrm{e}^{i\omega t}$ 。

对于平面问题， $E_c = (0, 0, E_c), J = (0,\ 0,\ J\mathrm{e}^{j\omega t})$，且磁场强度

$$\boldsymbol{H} = (H_x, H_y, 0) = -\frac{1}{j\mu\omega} \nabla \times E_c,$$

则 E_c 的标量方程变为：

$$-\nabla \cdot \left(\frac{1}{\mu} \nabla E_c\right) + (j\omega\sigma - \omega^2\varepsilon) E_c = 0$$

在 PDE Specification 对话框中输入 PDE 参数，包括角频率ω，磁导率μ，电导率σ 和介电系数ε。

与本模式相关的边界条件为 Dirichlet 边界条件（在边界上指定电场强度 E_c）和 Neumann 条件（指定 E_c 的正导数）。这等于指定磁场强度 $\boldsymbol{H}$ 的切向分量：

$$H_t = \frac{j}{\omega} n \cdot \left(\frac{1}{\mu} \nabla E_c\right)$$

电场强度 $\boldsymbol{E}$ 可以求得的量有电流密度 $\boldsymbol{J}=\sigma \boldsymbol{E}$ 和磁感应强度

$$\boldsymbol{B} = \frac{j}{\omega} \nabla \times E$$

【例 29-4】 本例演示圆形断面、带交流电的铜线的趋肤效应。铜的电导率为 57×10^6，流通率为 1，$\mu=4\pi\times10^{-7}$。在频率为 50Hz 时，$\omega^2\varepsilon$ 项可以忽略。

由于感应现象，导线内部的电流密度比外表面的小，外表面处 $J_s=1$。电场为 Dirichlet 条件：$E_c=1/\sigma$。本例中可以获得解析解：

$$\boldsymbol{J} = J_s \frac{J_0(k_r)}{J_0(kR)}$$

式中， $k = \sqrt{j\omega\mu\sigma}$ ，R 为导线的半径，r 为至中心线的距离，$J_0(x)$为第一类零阶贝塞尔函数。

打开 pdetool，然后选择应用模式 AC Power Electromagnetics，并建立几何模型。

画一个半径为 0.1 的圆，代表导线的横断面。然后选择边界模式定义边界条件。用 Select All 选项选择所有的边界，在 Boundary Condition 对话框中的 r 编辑框中输入 1/57E6，以定义

Dirichlet 边界条件。

打开“PDE Specification 对话框”，输入 PDE 参数，角频率 $\omega = 2 \times pi \times 50$，如图 29-6 所示。

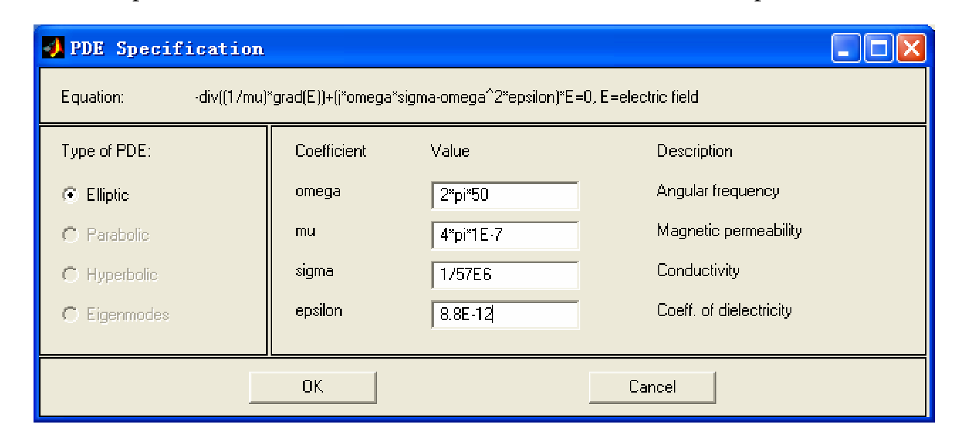


图 29-6　“PDE Specification”对话框

将网格初始化，求解方程。由于趋肤效应，导线表面的电流密度比导线内部的要高得多，这从电流密度 J 的三维图中可以清楚地看出。细化网格，可以获得更高精度的解。打开“Solve Parameters”对话框，选择 Adative mode 选项，将三角形网格的个数设为无穷(Inf)，细化的最大次数定义为 1。三角形选择方法为 Weight triangles。重新求解多次，每一次适应性求解器用最大误差来改善三角形面积。三角形的个数显示在命令行中。图 29-7 所示的网格为计算结果，其中有 833 个三角形。

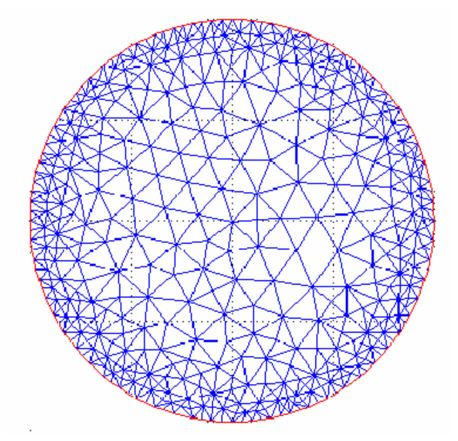


图 29-7　模型的适应性网格

交流电电磁学方程的解为复数解，如图 29-8 所示。图中显示了解的实数部分，但可以输出到主工作空间的解向量为复数解。也可以用自定义输入来以图形方式输出复数解的不同属性。imag(u)和 abs(u)是两个实例。

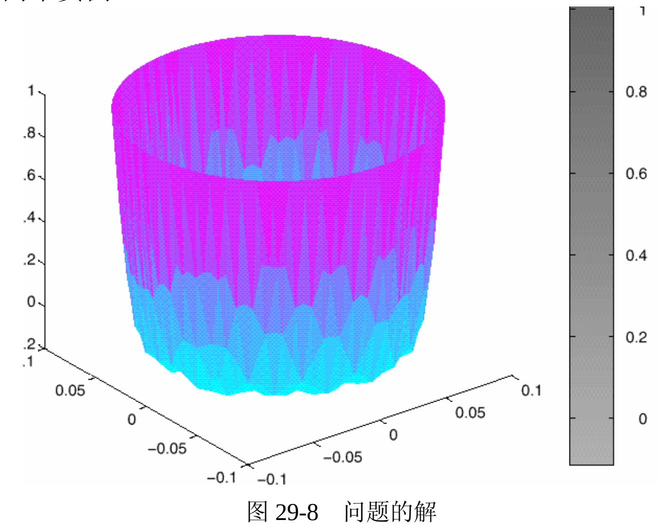


图 29-8　问题的解

趋肤效应是交流电产生的一种现象，降低交流电的频率可以减小趋肤效应。当接近直流电条件时，电流密度接近于均匀。

29.7　直流导电介质

对于诸接地板的电解和电阻计算，我们有电导率为σ的一种导电介质和一个平稳电流。该电流密度 $\boldsymbol{J}$ 与电场强度 $\boldsymbol{E}$ 的关系为 $\boldsymbol{J}=\sigma\boldsymbol{E}$。将连续方程$\nabla \cdot \boldsymbol{J}=Q$（其中 Q 为电流源），与电位 V 的定义相结合产生椭圆泊松方程：

$$-\nabla \cdot (\sigma \nabla V) = Q$$

仅有的两个偏微分方程（PDE）参数为电导率σ 和电流源 Q。

Dirichlet 边界条件指定该边界（通常为金属导体）的电位 V 的值。Neumann 边界条件要求该电流密度的法向分量（$n \cdot \nabla(V)$）为已知的。也可指定由 $n \cdot (\sigma \nabla(V)) + qV = g$ 定义的一个广义的 Neumann 条件，其中 q 可解释为接地板的薄膜电导。

电位 V、电场强度 E 和电流密度 J 都可以用图形表示。可视化感兴趣的量是电流线（J 的矢量场）和 V 的等位线。当σ 为各向同性时，等位线与电流线正交。

【例 29-5】　两个圆形金属导体放在一张被盐水浸湿的纸上。等位线可以通过伏特表用简易探针跟踪，电流线可以被强烈染色的离子跟踪。

本问题的物理模型可以用拉普拉斯方程来描述。

$$-\nabla \cdot (\sigma \nabla V) = 0$$

对于电势 V 和边界条件：

（1）V=1　左侧圆形导体；

（2）V=-1　右侧圆形导体；

（3）Neumann 边界条件 $\dfrac{\partial V}{\partial n} = 0$　外边界。

电导率σ=1（恒定）。

下面用 GUI 进行求解。

在命令窗口输入 pdetool 命令，选择应用模式 Conductive Media DC，建立几何模型。

首先画一个矩形，矩形四角坐标为(-1.2, -0.6), (1.2, -0.6), (1.2, 0.6) 和 (-1.2, 0.6)。该矩形代表纸片，然后添加两个圆，其半径为 0.3，圆心分别为(-0.6,0)和(0.6,0)。若矩形的标签为 R1，圆的标签为 C1 和 C2，则问题的二维分析域用集合公式

R1 -（C1+C2）

表示。

输入公式并单击∂Ω按钮，分解几何图形并输入边界模式。选择所有的外边界，在 Boundary Condition 对话框中输入 Neumann 边界条件。

对于左侧圆形导体边界，输入 Dirichlet 边界条件 V=1，对于右侧导体，输入 Dirichlet 条件 V=-1。

第 2 步，打开 PDE Specification 对话框，在编辑框中输入 0，作为电流源 q 的值，电导率的默认值为 1，不必改变。

将网格初始化，细化两次，微调网格一次以改善网格质量。

单击PDE按钮，求解 PDE。所得到的电位沿 Y 轴为零，对本问题它是一条反对称垂线。

用等值线图描绘电流密度 J 的绝对值，用箭头描绘矢量场使电流密度可视化，电流从正电位的导体流向负电位的导体，如图 29-9 所示。

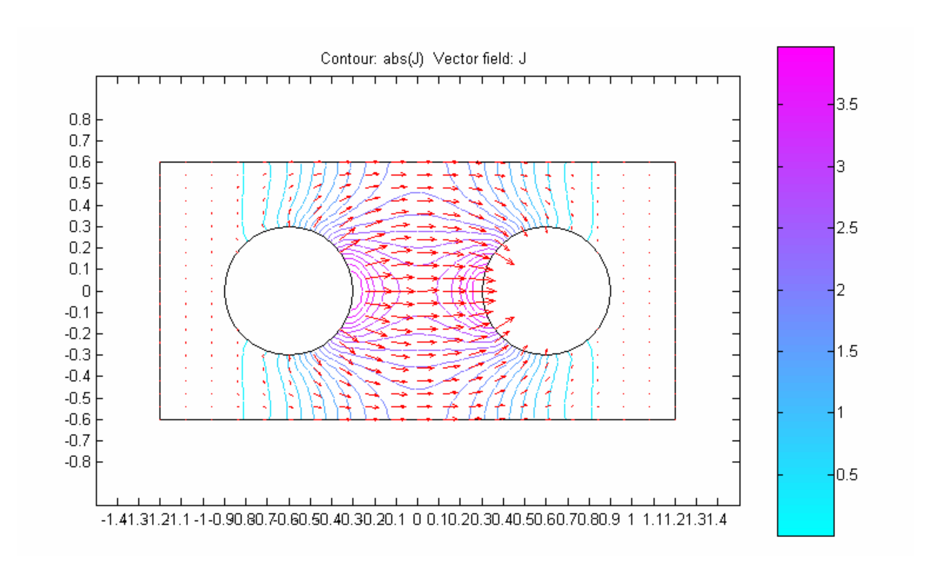


图 29-9　电流矢量图与电流密度等值线图

29.8　热传导

热传导方程为抛物线型 PDE 问题：

$$\rho C\frac{\partial T}{\partial t}-\nabla\cdot(k\nabla T)=Q+h\cdot(T_{ext}-T)$$

它描述平面轴对称问题的热传导方程，用到下面的参数：

（1）密度ρ；

（2）热容 C；

（3）热传导系数（热导率）k；

（4）热源 Q；

（5）对流热传递系数 h；

（6）外部温度 T_{ext}；

分项 h（T_{ext}-T）是与周围传热的模型，它对于薄的冷板中的热传递模型的计算是有用的。

对于稳定状况，可得到椭圆型热传导方程：

$$-\nabla\cdot(k\nabla T)=Q+h(T_{ext}-T)$$

边界条件可以是 Dirichlet 类型的，指定边界上的温度，或者是 Neumann 类型的，可以指定热感应强度 $\boldsymbol{n}\cdot(\boldsymbol{k}\nabla(\boldsymbol{T}))$。也可以用广义 Neumann 边界条件方程：$n\cdot(k\nabla(T))+qT=g$。其中 q 为热传递系数。

可以用图形方式输出温度、温度的梯度和热感应强度。图形选项包括 isotherms（等温线）

和 heat flux vector field plots（热流向量场图）。

【例 29-6】 在下面的例子中,求解一个不用材料参数的传热问题.

问题的二维分析域由一个嵌有金刚石(用一个旋转 45° 的方形表示)的方形组成。大方形由热传导系数为 10，密度为 2 的材料组成，金刚石形的区域包括一个量为 4 的均匀热源，它的热传导系数为 2，密度为 1。两个区域的热容都为 0.1。

使用 GUI 求解，其步骤如下。

打开图形用户界面，选择应用模式 Heat Transfer，绘制几何模型。

X 轴和 Y 轴的约束为[-0.5 3.5]，在 Options 菜单中选择 Axis Equal 选项。方形区域的角点的坐标为(0, 0), (3, 0), (3, 3)和(0, 3)。金刚石形状的区域角点坐标为(1.5, 0.5), (2.5, 1.5), (1.5, 2.5)和(0.5, 1.5)。

在所有的外边界上，温度保持为零，所以不必改变默认边界条件。

下一步定义 PDE 参数：双击两个区域，然后输入参数。

选择 Parabolic 选项，确定求解 PDE 的类型。

在大方形区域中，输入密度 2，热容 0.1 和热传导系数 10。

没有热源，故将其设为零，在金刚石形区域中，输入密度 1，热容 0.1 和热传导系数 2。在热源编辑框中输入 4，将对流热传递系数 h 设为 0。

由于是求解动态 PDE，必须定义初值和希望求解 PDE 的时间。在 Solve 菜单中选择 Parameters…选项，打开 Solve Parameters 对话框。本问题的动态过程十分迅速—温度在 0.1 个时间单位内即达到稳定状态。但可以捕获过程中感兴趣的部分进行显示，例如，输入 logspace(-2, -1, 10)作为时间向量，求对应的解。它给出 0.01 和 0.1 之间的以 10 为底对数算法的间隔数。温度的初值设为 0。若边界条件和初值不同，问题的公式包含不连续性。

图 29-10 为最终解的色谱图。图 29-11(a)～(d)为动态过程中得到的 4 张快照图。

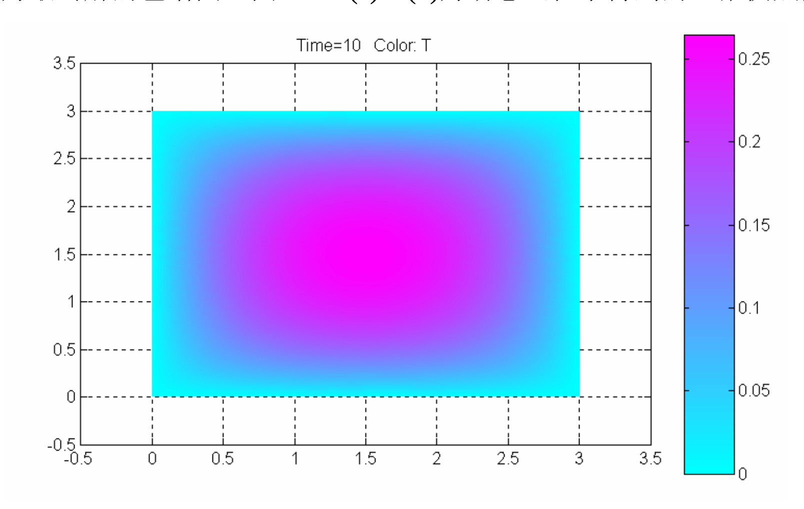
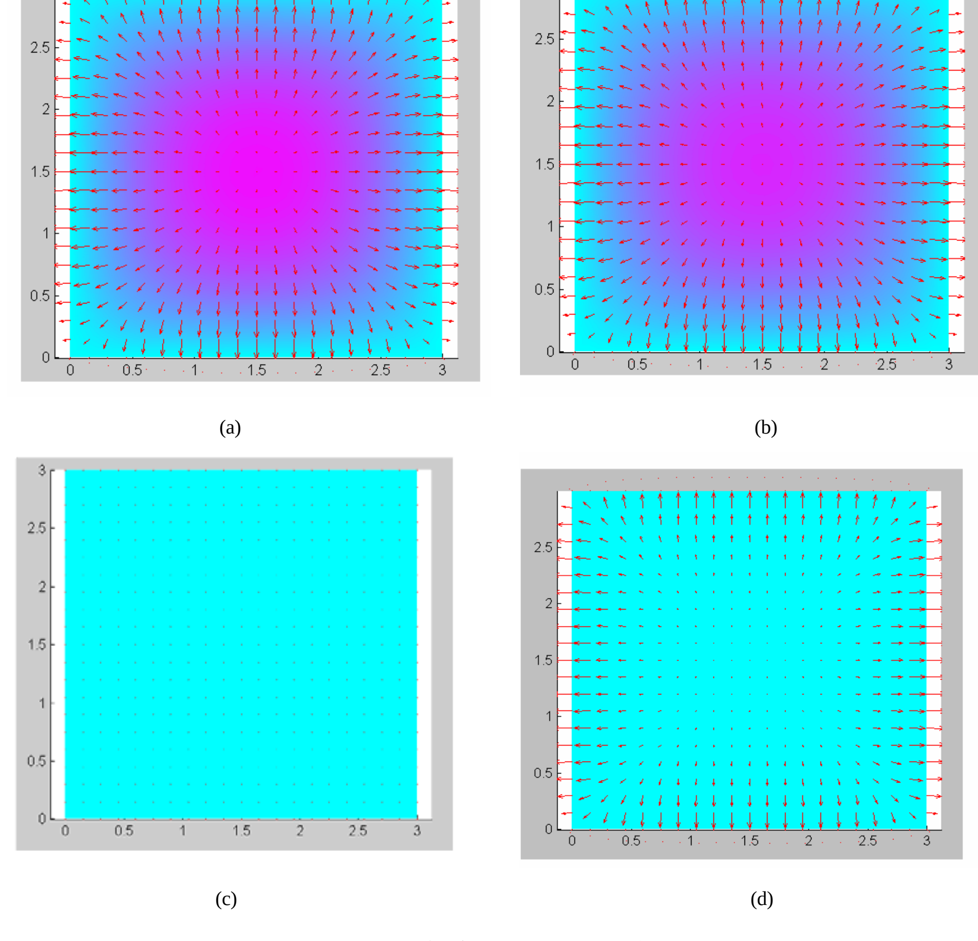


图 29-10 问题解的色谱图

求解 PDE。默认时，图形显示最后时间的温度分布。温度的动态行为可视化的一个最佳方法是激发问题的解。激发时，打开 Height(3-D plot)选项激活三维图形绘制，也可以选择 Plot in x-y grid 选项。用方形网格代替三角形网格可以明显加快激发过程。

可以用等值线图来绘制等温线，用箭头绘制热流量向量场图。



(a) (b)

(c) (d)

图 29-11　色谱图叠加矢量图的快照

29.9　扩散问题

热传导是一个扩散过程，可描写为

$$\frac{\partial C}{\partial t} - \nabla \cdot (\boldsymbol{D}\nabla C) = Q$$

式中，C 为热容，D 为扩散系数，Q 为体源。

扩散过程可能不均匀，其中 $\boldsymbol{D}$ 为 2×2 的矩阵。

边界条件可以是 Dirichlet 类型，其边界上的热容是稳定的，或者 Neumann 类型，其中 $n \cdot (\boldsymbol{D}\nabla(C))$ 是指定的，也可以指定广义 Neumann 条件，由公式 $n \cdot (\boldsymbol{D}\nabla(C)) + qC = g$ 定义，其中 q 为传热系数。

可以利用“Plot Selection”对话框来实现热容及其梯度和热流量的图形化输出。

第四篇　样条工具箱

第 30 章　样条工具箱及样条曲线简介

样条曲线在工程实践与科学应用中有着广泛的应用。例如，试验、统计数据如何用曲线表示，设计、分析、优化的结果如何用曲线表示等，几乎各个领域都要用到样条曲线来对数据进行处理。

一般地，在工程实践与科学应用中经常遇到的绘制样条曲线的要求有以下几种。

（1）由试验或观测得到了一批数据点，要求用一个函数近似地表明数据点间的函数关系，并画出函数的样条曲线，这类问题称为样条曲线拟合问题。

（2）由试验、观测或计算得到了若干个离散点组成的点列，要求用光滑的样条曲线把这些离散点联结起来，这类问题称为样条曲线插值问题。样条曲线插值与样条曲线拟合不同，拟合并不要求样条曲线通过全部的数据点。

（3）在样条曲线形状设计中，给定了折线轮廓，要求用样条曲线逼近这个折线轮廓，这类问题称为样条曲线逼近问题。

上述各类问题都要求画出样条曲线，它们都要求首先找出或构造出样条曲线的方程，再根据曲线方程画出样条曲线。根据方程画样条曲线一般是先计算出样条曲线上一系列适当靠近的点，然后依次将这些点用直线连起来，得到一条由折线表示的近似曲线。只要这些点靠得足够近，看起来就是一条光滑的样条曲线。上述构造的样条曲线方程的表示形式有显式、隐式和参数表示等几种。其中，最常用的是参数形式。利用参数形式，计算样条曲线的切矢量、法矢量、曲率和挠率等都非常方便。

正因为样条曲线的重要性与广泛应用，MATLAB 专门提供了样条工具箱，以使用户能够方便地处理各种数据，生成样条曲线。在 MATLAB 的样条工具箱中，包含了非常多的函数。这些函数按主题分类如下。

- 三次样条曲线类函数——这类函数主要用于在各种条件下生成三次样条曲线。
- 分段多项式样条曲线类函数——这类函数主要用于生成分段多项式样条曲线。
- B 样条曲线类函数——这类函数主要用于在各种条件下生成 B 样条曲线。
- 有理样条曲线类函数——这类函数主要用于在各种条件下生成有理样条曲线。
- 操作器类函数——这类函数主要对样条函数进行各类基本的操作，如计算样条函数值、画出样条曲线等。
- 断点及扭结点处理类函数——这类函数主要用于对生成样条曲线的数据点，即断点或扭结点进行处理（为了读者阅读方便，本章中的断点或扭结点与节点概念等同）。
- 解线性方程组类函数——这类函数主要用于求解线性方程组。
- 样条曲线的 GUI 类函数——这类函数具有图形用户界面，通过它们用户可以方便地画

出各种样条曲线。

下面，我们对上述各类函数做一个详细的介绍，以使读者能完全掌握 MATLAB 功能强大的样条工具箱中的各类函数，并能熟练地应用各个函数解决工程实践与科学应用中碰到的问题。

第 31 章　三次样条曲线

31.1　基本原理

三次样条曲线中，两节点（断点）之间的曲线段由一个三次多项式拟合生成，整个样条曲线由一段一段的三次多项式曲线组成，各段三次多项式具有不同的系数。生成三次样条曲线的关键是要寻找三次多项式，以逼近每对节点间的曲线。因为两点只能决定一条直线，而在两点间的曲线可用无限多的三次多项式近似。因此，三次多项式还应满足以下条件，即两相邻段在其联结点处（即节点）应二阶连续。也就是说，在节点处的函数值、一阶导数、二阶导数都应连续，这就较好地确定了所有内部三次多项式。

设有一个节点列 P_i（x_i，y_i），i=0~n，要构造三次样条曲线。

记样条函数 $S(x)$在 x=x_i 节点处的函数值、一阶导数、二阶导数分别为：

$$S(x_i) = y_i$$

$$S'(x_i) = m_i$$

$$S''(x_i) = M_i$$

在每个小区间[x_{i-1} x_i]上，$S(x)$的二阶导数是线性的，即

$$S''(x) = M_{i-1}\frac{x_i - x}{h_i} + M_i\frac{x - x_{i-1}}{h_i}$$

式中，h_i=x_i−x_{i-1}。

将 S''(x)积分二次，得到：

$$S'(x) = -M_{i-1}\frac{(x_i - x)^2}{2h_i} + M_i\frac{(x - x_{i-1})^2}{2h_i} + k \qquad (31\text{-}1)$$

$$S''(x) = m_{i-1}\frac{(x_i - x)^3}{6h_i} + M_i\frac{(x - x_{i-1})^3}{6h_i} + kx + p \qquad (31\text{-}2)$$

式中，k 与 p 为积分常数。

把小区间首末两端的坐标 $x = x_{i-1}$， $S(x) = y_{i-1}$ 和 $x = x_i$， $S(x) = y_i$ 分别代入式(31-2)，得

$$y_{i-1} = M_{i-1}\frac{h_i^2}{6} + kx_{i-1} + p \qquad (31\text{-}3)$$

$$y_i = M_i\frac{h_i^2}{6} + kx_i + p \qquad (31\text{-}4)$$

（31-3），（31-4）两式联立求解得

$$k = \frac{y_i - y_{i-1}}{h_i} + \frac{h_i(M_i - M_{i-1})}{6} \qquad (31\text{-}5)$$

$$p = (\frac{y_i}{h_i} - \frac{h_i}{6}M_{i-1})x_i - (\frac{y_i}{h_i} - \frac{h_i}{6}M_{i-1})x_{i-1} \qquad (31\text{-}6)$$

将式(31-5)、（31-6）两式代入（31-1)、（31-2）得

$$S'(x) = -M_{i-1}\frac{(x_i - x)^2}{2h_i} + M_i\frac{(x - x_{i-1})^2}{2h_i} + \frac{y_i - y_{i-1}}{h_i} - \frac{h_i(M_i - M_{i-1})}{6} \quad (31\text{-}7)$$

$$S(x) = M_{i-1}\frac{(x_i - x)^3}{6h_i} + M_i\frac{(x - x_{i-1})^3}{6h_i} + \left(\frac{y_{i-1}}{h_i} - \frac{h_i}{6}M_{i-1}\right)(x_i - x) + (\frac{y_i}{h_i} - \frac{h_i}{6}M_i)(x - x_{i-1}) \quad (31\text{-}8)$$

在各中间点 p_i，i=1~n，第 i 段末端 x=x_i 处，

$$S''(x_i - 0) = \frac{h_i}{6}M_{i-1} + \frac{h_i}{3}M_i + \frac{y_i - y_{i-1}}{h_i}$$

在第 i+1 段始端 x=x_i 处，

$$S''(x_i + 0) = -\frac{h_{i+1}}{3}M_i - \frac{h_{i+1}}{6}M_{i+1} + \frac{y_{i+1} - y_i}{h_{i+1}}$$

各中间点的一阶导数连续，即 $S'(x_i\text{-}0)= S'(x_i\text{+}0)$，经过化简得

$$\mu_i M_{i-1} + 2M_i + \lambda_i M_{i+1} = d_i \quad (31\text{-}9)$$

式中，$\mu_i = \dfrac{h_i}{h_i + h_{i+1}}$，$\lambda_i = \dfrac{h_{i+1}}{h_i + h_{i+1}}$，$d_i = \dfrac{6}{h_i + h_{i+1}}(\dfrac{y_{i+1} - y_i}{h_{i+1}} - \dfrac{y_i - y_{i-1}}{h_i})$

当 i 取 1, 2, ⋯, n–1 时，可得到 n–1 个形如式（31-9）的方程。但未知数的二阶导数 M_i 却有 n+1 个，即 $M_0, M_i, \cdots, M_n$。要惟一地确定解，必须再附加两个方程。通常按实际问题的具体情况，在样条两端，即 p_0 和 p_n 处给出约束条件，这种条件称为边界条件。常用的边界条件有：

（1）给定两端的斜率，即 $m_0 = y_0$ 和 $m_n = y_n$

以 x=x_0，i=1 代入式（31-7)，得

$$2M_0 + M_1 = \frac{6}{h_i}(\frac{y_i - y_0}{h_1} - y_0') \quad (31\text{-}10)$$

以 x=x_n，i=n 代入式（31-7)，得

$$M_{n-1} + 2M_n = \frac{6}{h_n}(y_n' - \frac{y_n - y_{n-1}}{h_n}) \quad (31\text{-}11)$$

式（31-9）和附加的两个方程（31-10)、（31-11）合在一起得到有定解的线性方程组。写成矩阵形式为

$$\begin{bmatrix} 2 & \lambda_0 & & & & & \\ \mu_1 & 2 & \lambda_1 & & 0 & & \\ & \mu_2 & 2 & \lambda_2 & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \mu_n - 2 & 2 & \lambda_n - 2 & & \\ & 0 & & \mu_n - 1 & 2 & \lambda_n - 1 \\ & & & & \mu_n & 2 \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ M_2 \\ \vdots \\ M_{n-2} \\ M_{n-1} \\ M_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ \vdots \\ d_{n-2} \\ d_{n-1} \\ d_n \end{bmatrix} \quad (31\text{-}12)$$

式中，λ_0=1，　$d_0 = \dfrac{6}{h_1}(\dfrac{y_1 - y_0}{h_1} - y_0')$

μ_n=1，　$d_n = \dfrac{6}{h_n}(y_n' - \dfrac{y_n - y_{n-1}}{h_n})$

（2）给定两端的二阶导数，即 $M_0= y''_0$ 和 $M_n= y''_n$

这可以写成

$$2M_0 + 0 \times M_1 = 2y''_0$$
$$0 \times M_{n-1} + 2M_n = 2y''_n$$

此时，式（31-12）中的λ_0=0，d_0=2 y''_0；μ_n=0，d_n=2 y''_n。

（3）使第 1 个节点的二阶导数与第 2 个节点的二阶导数相等，即 M_0=M_1，同时，使最后一个节点的二阶导数与倒数第 2 个节点的二阶导数相等，即 M_{n-1}=M_n。此时，式（31-12）中的λ_0=-2，d_0=0； μ_n=-2，d_n=0。实际上，这就是抛物线边界条件。

满足上面的边界条件，而且各节点处的函数值、一阶导数值、二阶导数值都连续，就可以生成三次样条曲线了。

31.2　三次样条曲线的生成

MATLAB 样条工具箱中提供的生成三次样条曲线的函数如表 31-1 所示。

表 31-1　三次样条曲线类函数

函 数 名 称	简 单 介 绍
csapi	插值生成三次样条函数
csape	生成给定约束条件下的三次样条函数
csaps	平滑生成三次样条函数
cscvn	生成一条内插参数的三次样条曲线
getcurve	动态生成三次样条曲线

1．csape 函数

利用 cscape 函数生成给定数组 *x*、*y* 下，满足 *s*(*x*(*j*))=*y*(*j*)的三次样条函数，其调用格式为：

- pp = csape(x,y)
- pp = csape(x,y,conds,valconds)

两端点的约束条件由参数 conds 与 valconds 确定。其中，conds 是一个字符串，可取以下值：'complete'或'clamped', 'not-a-knot', 'periodic', 'second', 'variational'，它们分别代表下面的意思。

① complete：给定第 1 个点与最后一个点（即两端点）的切矢量（一阶导数）。此时，当参数 valconds 也给出时，用做默认值。

② not-a-knot：使第 2 个节点与倒数第 2 个节点成为不活动节点，也就是说忽略这两个节点。此时，忽略参数 valconds 的作用。

③ periodic：给定第 1 个点与最后一个点的一阶与二阶导数。

④ second：给定最后一个点的二阶导数。当参数 valconds 也给出，且其值为默认值[0 0]时，相当于 variational。

⑤ variational：使最后一个点的二阶导数等于零，此时忽略参数 valconds 的作用。

⑥ default：用所给定的 4 个点通过 Lagrange 插值法，生成三次样条曲线。

另外，也可对取值区间的两端点给出不同的约束条件，这时 conds 为矩阵形式，且对每个 conds(j)需给出不同的值，以对两端点确定不同的约束条件。

定义如下。$D^i s$ 表示第一个点(j=1)的 valconds(j)参数的值；如果 conds(j)=i，表示最后

一个点(j=2) 的 valconds(*j*) 参数的值，则此处的 i=1 或 2。表 31-2 是 conds 与 valconds 的可选值。

当 j=1 或 2 时，e=a(b)是第一个点（j=1）或最后一个点（j=2）的值，则在 Lagrange 插值条件下，p 是在给定点 e 以及与 e 最近的三个点插值所得的三次样条插值函数。

如果参数 conds(j) 没有明确给出，则 conds(j)默认为 1，同时，valconds(j) 参数设为默认值。

valconds(j) 的默认值是 conds(j)=1 或 0 时，三次样条插值最近的 4 个点的一阶导数值。即，csape(x,y)表示在 Lagrange 约束条件下的三次样条插值。

表 31-2　参数 conds 与 valconds 的可选值

lagrange	Ds(e)=Dp(e)	默认值
clamped	Ds(e) = valconds()	conds() = 1
variational	$D^2s(e) = 0$	conds() = 2, valconds(j) = 0
periodic	$D^rs(a) = D^rs(b)$,r = 1,2	conds() = [0 0]
curved	$D^2s(e)$ = valconds()	conds() = 2

csape(x,y,[2 2]), csape(x,y,’v’) 为 variational 约束条件下的三次样条插值。

而 csape([-1 1], [-1 1], [1 2], [3 6]) 表示 p(-1)= -1、Dp(-1)=3、p(1)=1、D^2p(1)=6，即 p(x)=x^3 下的三次样条插值。

csape([-1 1], [-1 1]) 表示 p(-1)= -1、p(1)=1，即 p(x)=x 下的三次样条插值。

当然，也可以把参数 valconds 包括在参数 y 内。特别地，如果 size(y)==[d,ny]，其中 ny=length(x)+2，此时 valconds 为与 x(i)，i=1:length(x) 相对应的 y(:,[1 end])与 y(:,i+1)。

若 x 是一个包含有 m 个元素的数组 [x1…xm]，则相应地，y 也是一个包含有 m 个元素的数组。若函数是矢量形式，则 y 是一个有 m+1 个元素的数组。同时，conds 是一个包含有 m 个元素的数组。此时，valconds 成了 y 的一部分。

【例 31-1】　下面是一个多元矢量函数的例子。

```
>>x = 0:4; y=-2:2; s2 = 1/sqrt(2);
clear v
v(3,:,:) = [0 1 s2 0 -s2 -1 0].'*[1 1 1 1 1];
v(2,:,:) = [1 0 s2 1 s2 0 -1].'*[0 1 0 -1 0];
v(1,:,:) = [1 0 s2 1 s2 0 -1].'*[1 0 -1 0 1];
sph = csape({x,y},v,{'clamped','periodic'});
values = fnval(sph,{0:.1:4,-2:.1:2});
surf(squeeze(values(1,:,:)),squeeze(values(2,:,:)),...
squeeze(values(3,:,:))); axis equal
```

这是一个二元三次样条插值函数，计算结果是一个球，如图 31-1 所示。

上例中，也可以用命令 fnplt(sph) 代替 fnval 与 surf 命令来画图。此处的 v 是三维数组，v(:,i,j)与 (x(i),y(j)) 相对应，i=1:5, j=1:5。为了与 conds（1），即 clamped 相对应，size(v,2)是 7，而不是 5，v(r,:,j)的第一个与最后一个点与两端点的一阶导数（即切矢量）相对应。

以前的 MATLAB 版本中列出的约束条件可以被现在的约束条件代替，例如，实现下面的约束条件：

$$\lambda(s) = \alpha Ds(e) + b\,D^2 s(e) = c$$

式中，*a*, *b*, *c* 为标量，*e*=*x*(1)。

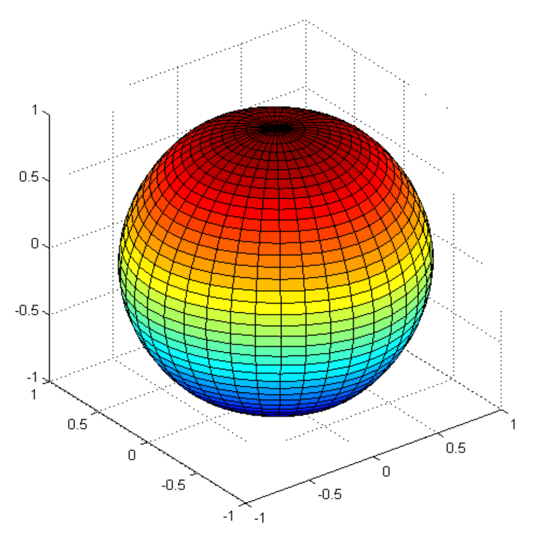


图 31-1　三次样条曲面图

使用默认的约束条件计算给定点处的三次样条插值函数 s1，同时计算在零点和约束条件 *e* 下的三次样条插值函数 s0，最后获得期望的三次样条插值函数：

$$s = s_1 + ((c - \lambda)(s_1)) / \lambda(s_0)s_0$$

下面是详细的实现过程，这里为了不对 s1 与 s0 进行微分，删除了分段多项式中的 s1 与 s0 两段。

```
>>x=2*pi*[0 1 0.1 0.2 0.9];y=cos(x);
pp1 = csape(x,y);
dp1 = fnder(fnbrk(pp1,1));
pp0 = csape(x,zeros(size(y)),[1,0],[1,0]);
dp0 = fnder(fnbrk(pp0,1));
e = x(1); a=4;b=3;c=2;
lam1 = a*fnval(dp1,e) + b*fnval(fnder(dp1),e);
lam0 = a*fnval(dp0,e) + b*fnval(fnder(dp0),e);
pp = fncmb(pp0,(c-lam1)/lam0,pp1);fnplt(pp）
```

结果如图 31-2 所示。

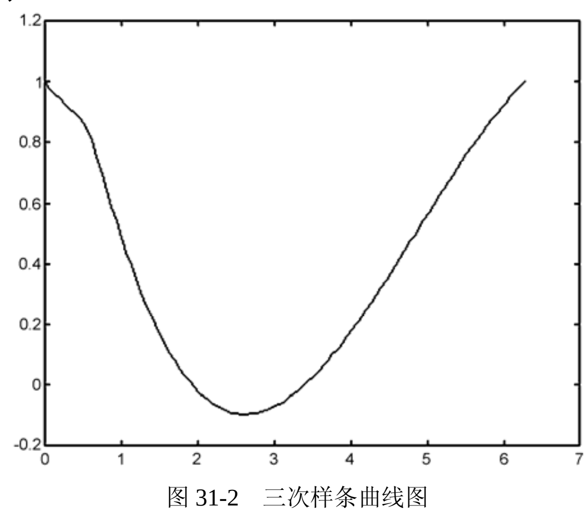


图 31-2　三次样条曲线图

注意：如果 x 数组不是单调递增的，则必须对 x 与 y 重新进行排列，以使它们符合单调递增。另外，若 y 是向量形式，则 valconds(:,j)，j=1:2 也必须是同样长度的向量形式。

2．csapi 函数

给定节点数组 x，利用 csapi 函数求出满足 $s(x(j))=y(j)$ 的三次样条曲线 s。此函数的功能与 csape 函数在 conds='not-a-knot' 约束条件下的功能相同，其调用格式为：

- values = csapi(x,y,xx)　首先生成满足 $s(x(j))=y(j)$ 的三次样条曲线 s，然后求出在给定节点数组 xx 下的函数值 $s(xx)$，结果存放在输出数组 values 中。
- pp = csapi(x,y) 返回满足 $s(x(j))=y(j)$ 的三次样条曲线 s，结果存放在 pp 中。pp 可以作为 fnval、fnder 等操作器类函数的输入参数。

如果 x 是一个包含有 m 个数组元素 $x_1…x_m$ 的数组，而每个数组元素又包含其他数组元素，即形如{{a1 a2 a3},{a4 a6},{a7 a9 a10 a11}}的数组，则数组 x 的每个元素的长度分别为 $n_1…n_m$。此时，要求 y 也是一个数组，数组长度为$[n_1,…,n_m]$，输出结果为一 m 元三次样条函数。如果只有两个输入参数，则返回的是分段多项式形式的样条函数 pp；如果输入参数是 3 个，则返回数组 xx 的样条函数值 $s(xx)$，如果 xx 数组有 m 个元素，则返回的也是一个包含有 m 个元素的数组。

在舍入误差允许范围内，如果 x 至少有 4 个元素，则 pp=csapi(x,y)与下面所示返回的结果相同：

```
n = length(x);
pp = fn2fm(spapi(augknt(x([1 3:(n−2) n]),4),x,y),'pp');
```

但是，如果计算三次样条函数时，没有把节点 $x(2)$与 $x(n-1)$考虑在内，返回的结果可能不同。

【例 31-2】　下面是一个二元双三次样条插值函数的例子。

```
>>x =.0001+[−4:.2:4]; y = −3:.2:3;
[yy,xx] = meshgrid(y,x); r = pi*sqrt(xx.^2+yy.^2); z = sin(r)./r;
bcs = csapi( {x,y}, z ); fnplt( bcs ), axis([−5 5 −5 5 −.5 1])
```

返回的图形是一顶墨西哥帽子，如图 31-3 所示。

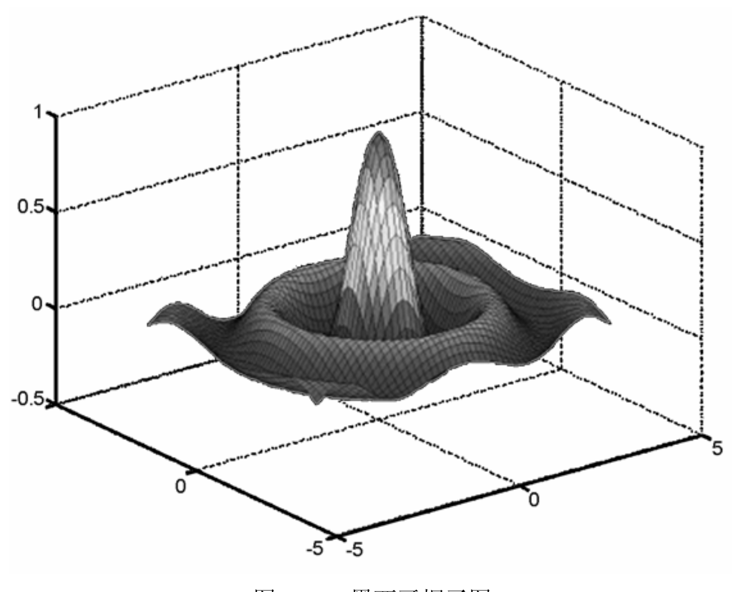


图 31-3　墨西哥帽子图

注意：若数组 x 不是单调递增，则要求 x 与 y 重新排列，以使它们符合单调递增。

3．csaps 函数

用 csaps 函数生成平滑的三次样条函数，其调用格式为：

- [values, p] = csaps(x, y, p, xx)
- [values, p] = csaps(x, y, p, xx, w)
- [pp, p] = csaps(x, y)
- [pp, p] = csaps(x, y, p)
- [pp, p] = csaps(x, y, p, [ ], w)

给定平滑参数 p（取值区间为[0 1]）和可选参数 w（权重），对数据 x，y 进行平滑处理，生成三次样条曲线。如果平滑参数 p 是负值，或者没有给出，该函数会自动选择有效的 p，此时 p 会作为第二个输出变量返回。

平滑三次样条曲线的方程如下：

$$p \cdot \sum_i w(i)((y)(i)-s(x(i)))^2+(1-p)\int(D^2 s)^2=0$$

式中，w=ones(size(x))，当输入参数 w 没有给出时，它就是 w 的默认值。当 p=0 时，用最小二乘法来平滑生成三次样条函数，当 p=1 时，相当于用命令 csapi（x,y, variational）来拟合生成三次样条函数。当 p 从 0 到 1 变化时，平滑样条曲线也随着从一端变化到另一端。p 的步长一般为 $1/(1+h^3/6)$，式中的 h 为横坐标的平均间距。对于均匀间距，p 在 $1/(1+h^3/60)$与 $1/(1+h^3/0.6)$间变化。

csaps(x,y,p,xx)返回给定点 xx（xx 可以是一个数组）的三次样条函数值 $s(xx)$（与之相对应，$s(xx)$也可以是一个数组）。

csaps(x,y,p)返回给定平滑参数 p，经过平滑处理的三次样条函数。因为选择合适的参数 p 是很困难的，所以建议用 spaps 函数来产生合适的 p，以便在允许的精度范围内产生最光滑的样条曲线。

当然，也可以对离散性很大的散点数据进行平滑处理。通过调用

```
values = csaps( {x1,...,xm}, y, p, xx, w)
```

或

```
pp= csaps( {x1,...,xm}, y, p, [], w )
```

就可以对散点数据进行平滑处理，获得平滑的样条曲线。如果函数 pp 是矢量形式，则数组 y 的长度为[d,length(x1),⋯,.length(xm)]；如果函数 pp 为标量形式，则数组 y 的长度为[length(x1),⋯,.length(xm)]。p 是标量或包含有 m 个元素的向量，xx 为 xx(:,j)或{xx1,⋯, xxm}，可求出这些零散节点的函数值。如果参数 w 也给出，则 w 为包含有 m 个元素的数组（w{i}为空，则用默认的权重代替）。

【例 31-3】

```
>>x=2*pi*[0 1 0.1 0.2 0.9];
y=cos(x);
csaps(x,y,1,[0.3 0.5 0.6])
ans =
0.9416   0.8740   0.8251
pp=csaps(x,y,1);
fnplt(pp)
```

结果如图 31-4 所示。

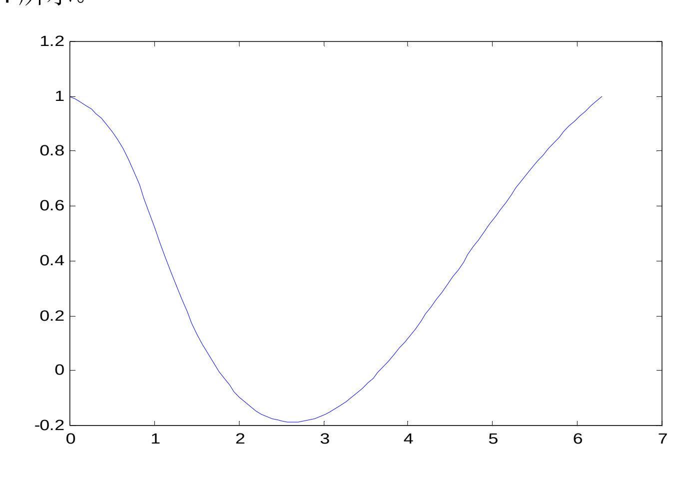


图 31-4　平滑生成的三次样条曲线图

注意：若 x 数组不是单调递增的，则要求 x 与 y 重新排列，以使它们符合单调递增。

4．cscvn 函数

该函数在 natural 或 periodic 约束条件下内插生成三次样条曲线，其调用格式为：

● curve = cscvn(points)　返回在给定点 points(:,j)，j=1:end 的参变量函数值（存放在向量 curve 中）。在第 j 个点的参变量值 t(j) 可通过下式求得

$$\sum_{i>j}\sqrt{\|\mathrm{points}(:,i+1)-\mathrm{points}(:,i)\|_2}$$

如果第 1 个点与最后一个点重合，则生成封闭的三次样条曲线。

【例 31-4】　下面的例子产生一条通过给定点的样条曲线。

```
>>points=[0 1 1 0 -1 -1 0 0 ;0 0 1 2 1 0 -1 -2];
fnplt(cscvn(points)); hold on,
plot(points(1,:),points(2,:),'o'), hold off
```

结果如图 31-5 所示。

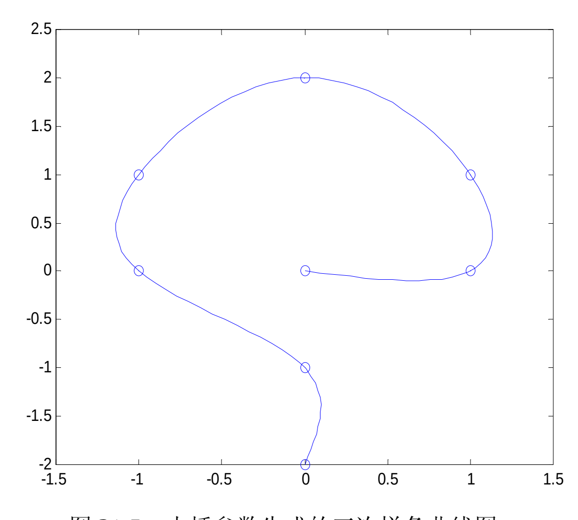


图 31-5　内插参数生成的三次样条曲线图

下面的例子利用函数 cscvn 生成一个“心形”图形，如图 31-6 所示。

```
>>c=fnplt(cscvn([0 .82 .92 0 0 -.92 -.82 0; .66 .9 0 -.83 -.83 0 .9 .66]));
fill(c(1,:),c(2,:),'r');  %利用 fill 函数，在样条曲线范围内充填红色
axis equal
```

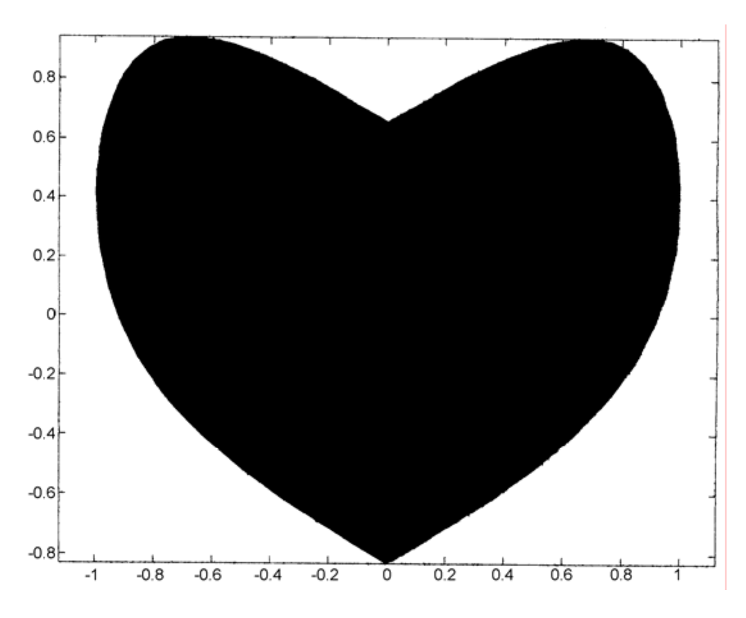


图 31-6　内插参数并充填生成的三次样条曲线图

5．getcurve 函数

用 getcurve 函数动态生成三次样条曲线，其调用格式为：

[xy, spcv] = getcurve

在 MATLAB 工作空间内输入 getcurve 函数后，首先会显示一个网格窗体界面，用户可用鼠标在窗体范围内随意地点一些点。随之，在窗体上出现联结这些点的一段段的直线。最后当用户在窗体范围外任意地方用鼠标单击一下后，就会画出通过这些点的三次样条曲线（此样条曲线是通过 cscvn 函数计算得到的）。如果用户在网格窗口内点的最后一个点与第一个点重合，则画出的是封闭的样条曲线。

【例 31-5】　用户在窗体范围内随意点击：

```
ans =
  Columns 1 through 7
-0.8710   -0.8111   -0.3733    0.2396    0.5392    0.2212   -0.2949
0.5965    -0.1170   -0.4795   -0.4737    0.1345    0.2573    0.5380
  Column 8
-0.4839   0.6667
```

所画的图形如 31-7 所示。

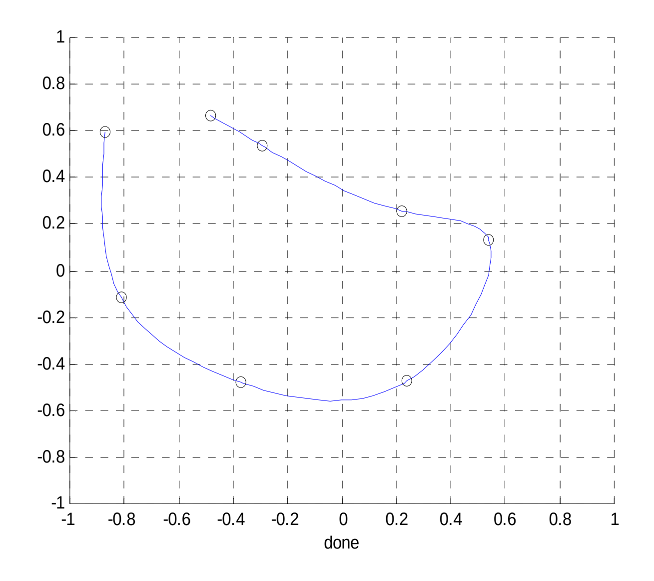


图 31-7　动态生成的三次样条曲线图

第 32 章　分段多项式（PP）样条曲线

32.1　基本原理

分段多项式样条曲线（即 PP 形式的样条曲线）中，两节点（断点）之间的曲线段是由多项式拟合生成的，整个样条曲线由一段段的分段多项式曲线组成，各段多项式具有不同的系数。生成分段多项式样条曲线的关键是寻找多项式，以逼近每对节点间的曲线。因此，在生成分段多项式样条曲线时，要求用户输入各分段的系数值，以此来拟合生成分段多项式样条曲线。

在 MATLAB 样条工具箱中，一个（单变量）分段多项式样条函数由它的节点数组 breaks 和多项式各段的系数数组 coefs 拟合生成。当样条曲线是二维或三维时，与此相对应的系数数组 coefs 也变成二维或三维向量。然而，为了与标准的向量运算一致，系数向量常按一维进行运算。另外，节点数组 breaks 要求必须严格单调增，即：

breaks(1)<breaks(2)<…<breaks(m+1)

其中，m 为分段多项式的段数。各段多项式的阶次可以不同，但最好具有相同的阶次 k。此时，若系数数组 coefs 的大小是[m,k]，则 coefs(j,:)是第 j 段分段多项式的系数。

分段多项式样条函数 f 就是由节点数组 breaks、系数数组 coefs、段数 m 和阶次 k 生成。节点的取值区间是[breaks(1)　breaks(m+1)]。此时，分段多项式样条函数可按如下方式表示：

f(t) = polyval(coefs(j,:),　t－breaks(j))

其中，breaks(j) ≤t < breaks(j+1)

polyval(a, x)是 MATLAB 中的函数，函数值通过下式计算获得：

$$\sum_{j=1}^{k} a(j)x^{k-j} = a(1)x^{k-1} + a(2)*x^{k-2} + ... + a(k)*x^{0}$$

当 t 不在区间[breaks(1)　breaks(m+1)]时，f(t)的第一段与最后一段分段多项式被扩展定义，即

f(t) = polyval(coefs(1,:),t-breaks(1))　，t < breaks(1)

一个分段多项式样条函数可以通过插值、拟合或从其他类型的样条函数中转换得到。这可以通过生成一些 M 文件来建立，输出结果为一行向量。但是，也可以通过如下方式来生成：

pp=ppmak(breaks,coefs)

例如，

pp=ppmak(－5:　－1,　－22:　－11)

上面的节点数组包含有 5 个元素，因此区间被分成四段。系数数组包含有 12 个元素。因此，我们可以确定该分段多项式样条函数的阶次为 3（即 12/4）。通过使用命令：

fnbrk(pp)

输出结果为：

```
breaks(1:m+1)
-5 -4 -3 -2 -1
coefficients(d*m,k)
-22 -21 -20
-19 -18 -17
-16 -15 -14
-13 -12 -11
pieces number m
4
order k
3
dimension d of target
1
```

从上面可知，通过函数 fnbrk（操作器类函数，后面有介绍），可以获得样条函数的有关信息。

下面，还有一些可对分段多项式样条函数进行操作的函数：

```
v=fnval(pp,x)
dpp=fnder(pp)
dirpp=fndir(pp,dir)
ipp=fnint(pp)
pj=fnbrk(pp,j)
pc=fnbrk(pp,[a b])
fnplt(pp)
sp = fn2fm(pp,'B-')
pr = fnrfn(pp,morebreaks)
```

上面的函数都是可以对样条函数进行操作的函数，本章后面将进行专门的介绍。

例如，可以通过如下操作来画出分段多项式样条曲线：

```
>> x = [-55:-5]/10;
pp=ppmak(-5:-1,-22:-11);
```

在图中画出竖线，

```
plot(x, fnval(pp,x),'-.');
breaks=fnbrk(pp,'b');
yy=axis;
hold on
for j=1:fnbrk(pp,'l')+1
plot(breaks([j j]),yy(3:4)),
end
```

最后，在图中画出第3段分段多项式样条曲线

```
plot(x,fnval(fnbrk(pp,3),x)),set(gca,'ylim',[-60 -10]),hold off
```

结果如图 32-1 所示。

注意，如果用一些数据可以很好地生成分段多项式样条曲线，则使用这些数据也可以生成 B 样条曲线，而且生成的效果会更好，因为分段多项式样条曲线实际上是 B 样条曲线线性组合的特例。

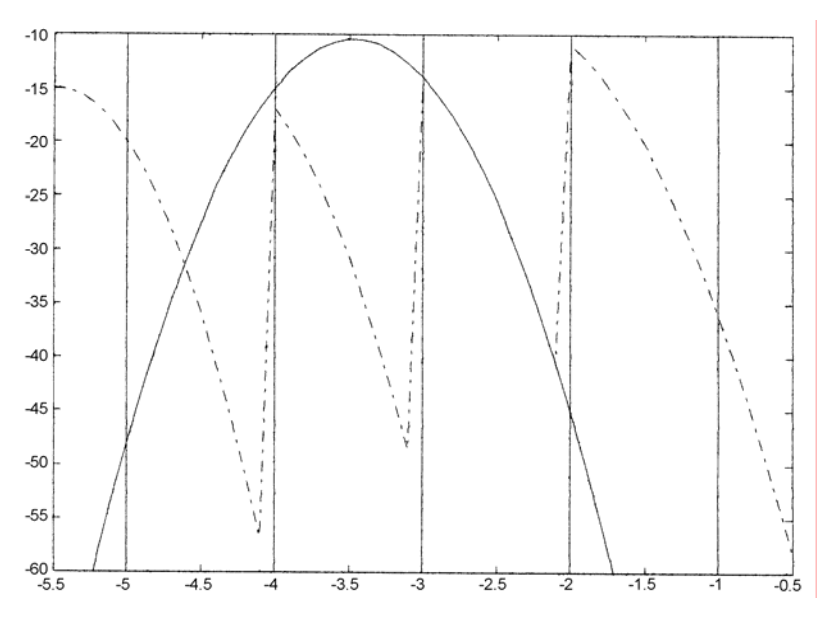


图 32-1　分段多项式样条曲线、第 3 段分段多项式样条曲线

32.2　分段多项式样条曲线的生成

MATLAB 的样条工具箱中提供的生成分段多项式样条曲线的函数如表 32-1 所示。

表 32-1　分段多项式样条曲线类函数

函 数 名 称	简 单 介 绍
pplst	显示关于生成分段多项式样条曲线的 M 文件
ppmak	生成分段多项式样条函数
ppual	计算在给定点处的分段多项式样条函数值

1．pplst 函数

利用该函数显示生成分段多项式样条曲线的 M 文件，其调用格式为：

- pplst

2．ppmak 函数

利用该函数生成分段多项式样条函数，其调用格式为：

- ppmak
- ppmak(breaks, coefs)
- pp = ppmak(breaks, coefs, d)

函数 ppmak 生成分段多项式样条函数。

如果用户只输入“ppmak”，则紧接着会出现提示，要求用户输入其他必须的参数（breaks 与 coefs 等）。通过这种方式，实际存储的这种形式样条函数的数据很容易被修改，而不会对与之有关的各个 M 文件有影响。输入参数 breaks 可用于决定 ppmak 命令生成的函数是一元函数还是多元函数。如果 breaks 参数是一个节点序列，即形如$\{a_1, a_2, \ldots, a_n\}$的数组（注意，breaks 数组中的元素必须是单调增的，而且要求第 1 个点与最后一个点不相等），此时生成的分段多

项工样条函数是一元函数。分段多项式样条函数的其他部分按下面决定：

（1）分段多项式段数 L 等于 breaks 数组的长度减 1，即 L=length(breaks)-1。相应地，各段的取值区间是[breaks(1)　breaks(L+1)]。

（2）函数的最高次幂（阶次）k 与维数 d 被决定如下。如果维数 d 没有明确给出，则 coefs(:,i*k+j)被认为是第 i+1 段分段多项式的第 j 个系数（要求该分段多项式的第 1 个系数最大、第 *k* 个系数最小，或者所有系数值都相等），然后维数 d 可从[d,kl]=size(coefs)得到，阶次 k=fix(kl/L)。如果维数 d 明确给出，则 coefs(i*d+j,:)被认为是第 i+1 段分段多项式的第 j 个系数。特别地，如果输入的系数数组 coefs 由 fnbrk 函数生成时，则维数 d 必须明确给出。如果节点数组 breaks 是一包含有 m 个元素 $x_1 \ldots x_m$ 的数组，而每个元素又包含其他的数组元素，即形如{{a1 a2 a3},{a4 a6},{a7 a9 a10 a11}}的数组，此数组的长度为 m，则分段多项式函数为 m 元函数。此时分段多项式样条函数的其他部分被决定如下。

① 第 i 个数组元素（它也是一个数组）的长度 L=length(breaks{i})-1，则相应的第 i 个数组元素（它也是一个数组）的取值区间是[breaks(i)(1) breaks(i)(end)]。

② 分段多项式函数各段的维数 d 与阶次 k 如果能直接从系数矩阵 coefs 中获得，则第 3 个输入参数 d 被忽略。如果 coefs 是一个长度为 m 的数组，则生成函数是标量形式，即 d=1，阶次 k=size(coefs)/L，同时系数矩阵被下式代替：coefs=reshape(coefs, [1, size(coefs)])。如果 coefs 是一个长度为 m+1 的数组，则 d=size(coefs, 1)，第 i 段多项式的阶次 k=size(coefs, i+1)/L(i)，i=1, m。

【例 32-1】

```
>>ppmak([0:2],[1:6])
ans =
   form: 'pp'
   breaks: [0 1 2]
   coefs: [2x3 double]
   pieces: 2
   order: 3
dim: 1
```

生成图形如 32-2 所示。

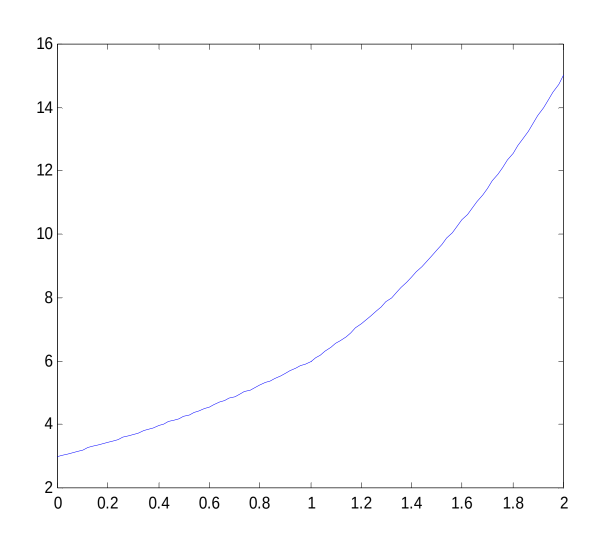


图 32-2　分段多项式样条曲线图

上例表示在区间[0 2]内生成一条分段多项式样条曲线，此曲线由两段多项式组成，曲线的阶为 3，生成的分段多项式函数为一元函数。此曲线有一个内部断点，函数在断点处是连续的，因为第一个分段多项式函数为

$$f(x) = x^2 + 2x + 3$$

第二个分段多项式函数为

$$f(x) = 4(x-1)^2 + 5(x-1) + 6$$

当函数是一元函数并且维数 d 没有明确给出时，d 等于系数矩阵 coefs 的行数。系数矩阵 coefs 的列数为分段多项式段数的乘方。例如，

```
ppmak([0:2],[1:3,4:6])
```

会导致错误，因为区间[0 2]表示有两段分段多项式，而系数矩阵中的列数不满足要求。修改了的

```
ppmak([0:1],[1:3,4:6])
```

表示抛物线曲线：

$$f(x) = (1,4)x^2 + (2,5)x + (3,6)$$

又如

```
>>ppmak([0:1,1:2],[1:3,4:6,5:7,6:8])
ans =
   form: 'pp'
   breaks: [0 1 1 2]
   coefs: [3x4 double]
   pieces: 3
   order: 4
dim: 1
```

图形如 32-3 所示。

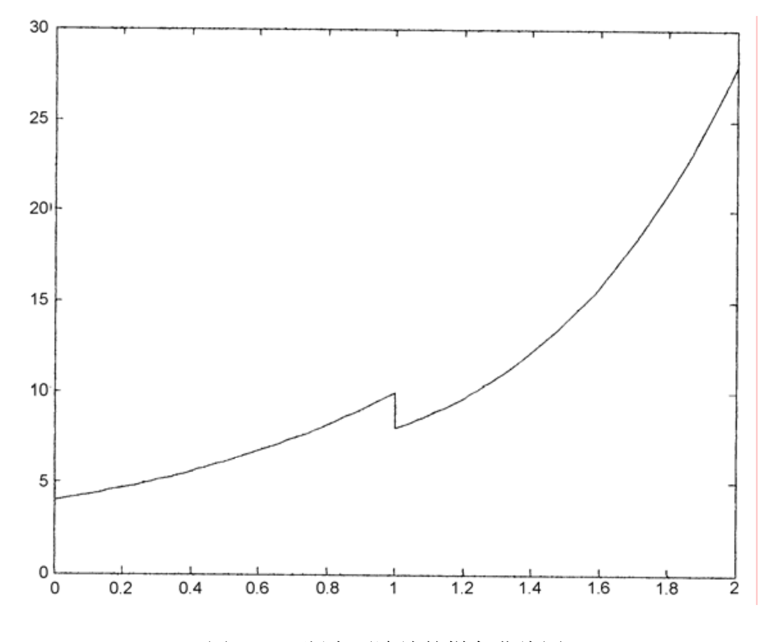


图 32-3 断点不连续的样条曲线图

3．ppual 函数

该函数计算给定点处的分段多项式样条函数值，其调用格式为：

● ppual[pp,x] 计算给定点 x（x 可以是一个矩阵或数组）处的分段多项式样条函数的函数值。如果分段多项式样条函数是一元 *d* 次函数，则函数值矩阵的大小是[d*m, n],其中[m, n]为 *X* 矩阵的大小。如果分段多项式样条函数是多元 d 次函数，则函数值矩阵的大小被决定如下：

（1）当 X 矩阵大小为[m,n]时，输出的函数值矩阵大小为[d*m, n]。

（2）当 d>1 并且 X 为[X1, X2…Xm]时，输出的函数值矩阵大小为[d, n1, n2…nm]。

（3）当 d=1 并且 X 为[X1, X2…Xm]时，输出的函数值矩阵大小为[n1, n2…nm]。

【例 32-2】

```
>>cs=ppmak([0:3,3:5],[1:3,4:6,5:7,6:8]);
ppual(cs,[1:2,3:4])
ans =
4     6     6     8
```

第 33 章　B 样条曲线

33.1　基本原理

B 样条曲线是最常用的样条曲线，它可以给出非常光滑的插值曲线，因此在数值逼近、常微分方程数值求解以及科学和工程计算中应用均相当广泛。

已知 n+1 个控制点 P_i (i=0,1,…,n)，称之为特征多边形的顶点，则把 n 次参数曲线段

$$P(t)=\sum_{i=0}^{n}P_l F_{l,n}(t) \tag{33-1}$$

叫做 B 样条曲线段。

式中

$$F_{l,n}(t)=\frac{1}{n!}\sum_{j=0}^{n-l}(-1)^j C_{n+1}^j(t+n-l-j)^n \tag{33-2}$$

而

$$C_{n+1}^j=\frac{(n+1)!}{j!(n+1-j)!}$$

一般地，由空间的 n+1 个控制点生成的 k 阶 B 样条曲线是由 L 段 B 样条曲线逼近而形成的，每个曲线段的形状由点列中 k 个顺序排列的点所控制；由不同节点向量构成的均匀 B 样条函数所描绘的形状相同，可看成是同一个 B 样条函数的简单平移。B 样条曲线的导数可用其低阶 B 样条基函数和顶点向量的差商的线性组合求出，这也是 k 阶 B 样条曲线段之间达到 K-2 次的连续性的原因。

工程实践中最常用的 B 样条曲线是低于三次的 B 样条曲线，高于三次的 B 样条曲线一般很少用。下面，我们着重对一、二、三次 B 样条曲线做一介绍。

1．一次 B 样条曲线

此时 n=1，由式（33-2）得

$$F_{0,1}(t)=-t+1$$
$$F_{1,1}(t)=t$$

代入（33-1）式，整理得

$$P(t)=(-t+1)P_0+tP_1$$

上式写成矩阵形式如下：

$$P(t)=[t\quad 1]\begin{bmatrix}-1 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}P_0\\ P_1\end{bmatrix}$$

如果继 P_0，P_1 之后还有一些点 P_2，P_3，…，则依次地每取两点，例如 P_1、P_2，P_2、P_3，…，都可以得到一段二次 B 样条曲线段，合起来就得到二次 B 样条曲线。

2．二次 B 样条曲线

此时 n=2，由式（33-2）得

$$F_{0,2}(t)=\frac{1}{2}(t-1)^2$$

$$F_{1,2}(t)=\frac{1}{2}(-2t^2+2t+1)$$

$$F_{2,2}(t)=\frac{1}{2}t^2$$

代入（33-1）式，整理得

$$P(t)=\frac{1}{2}[(P_0-2P_1+P_2)t^2+(-2P_0+2P_1)t+(P_0+P_1)] \tag{33-3}$$

这也可以写成矩阵形式：

$$P(t)=\frac{1}{2}[t^2 \quad t \quad 1]\begin{bmatrix}1 & -2 & 1\\ -2 & 2 & 0\\ 1 & 1 & 0\end{bmatrix}\begin{bmatrix}P_0\\ P_1\\ P_2\end{bmatrix}$$

当 t=0 时，$P(0)=\frac{1}{2}(P_0+P_1)$

当 t=1 时，$P(1)=\frac{1}{2}(P_1+P_2)$

这表明曲线段的两端点是二次 B 特征二边形两边的中点。

将式(33-3)对 t 求导，得

$$P'(t)=(t-1)P_0+(-2t+1)P_1+tP_2)$$

当 t=0 时，$P'(0)=P_1-P_0$

当 t=1 时，$P'(1)=P_2-P_1$

这表明曲线段两端点的切向量就是 B 特征二边形的两个边向量。

如果继 P_0、P_1、P_2 之后还有一些点 P_3、P_4…，则依次地每取三点，例如 P_1、P_2、P_3，P_2、P_3、P_4，…，都可以得到一段二次 B 样条曲线段，合起来就得到二次 B 样条曲线。

3．三次 B 样条曲线

此时 n=3，由式（33-2）得

$$F_{0,3}(t)=\frac{1}{6}(-t^3+3t^2-3t+1)$$

$$F_{1,3}(t)=\frac{1}{6}(3t^3-6t^2+4)$$

$$F_{2,3}(t)=\frac{1}{6}(-3t^3+3t^2+3t+1)$$

$$F_{3,3}(t)=\frac{1}{6}t^3$$

代入（33-1）式，整理得

$$P(t)=\frac{1}{6}[(-P_0+3P_1-3P_2+P_3)t^3+(3P_0-6P_1+3P_2)t^2+(-3P_0+3P_1)t+(P_0+4P_1+P_2)] \tag{33-4}$$

这也可以写成矩阵形式：

$$P(t)=\frac{1}{6}[t^3 \quad t^2 \quad t \quad 1]\begin{bmatrix}-1 & 3 & -3 & 1\\ 3 & -6 & 3 & 0\\ -3 & 0 & 3 & 0\\ 1 & 4 & 1 & 0\end{bmatrix}\begin{bmatrix}P_0\\ P_1\\ P_2\\ P_3\end{bmatrix}$$

当 t=0 时，$P(0)=\frac{1}{6}(P_0+4P_1+P_2)$

当 t=1 时，$P(1)=\frac{1}{6}(P_1+4P_2+P_3)$

将式(33-4)对 t 求导，得

$$P'(t)=\frac{1}{6}[3(-P_0+3P_1-3P_2+P_3)t^2+2(3P_0-6P_1+3P_2)t+(-3P_0+3P_2)] \quad (33\text{-}5)$$

当 t=0 时，$P'(0)=\frac{1}{2}(P_2-P_0)$

当 t=1 时，$P'(1)=\frac{1}{2}(P_3-P_1)$

将式(33-5)对 t 求导，得

$$P''(t)=\frac{1}{6}[6(-P_0+3P_1-3P_2+P_3)t+2(3P_0-6P_1+3P_2)] \quad (33\text{-}6)$$

当 t=0 时，$P''(0)=P_0-2P_1+P_2$

当 t=1 时，$P''(1)=P_1-2P_2+P_3$

如果 B 特征多边形添加一个顶点 P_4，则 P_0，P_1，P_2，P_4 决定下一段三次 B 样条曲线段。前一曲线段终点的位置、一阶、二阶导向量和下一曲线段始点的位置、一阶、二阶导向量仅与 P_1，P_2，P_3 有关，且都分别相等，这说明三次 B 样条曲线是二阶连续的。

在实际应用中，常希望所设计的三次 B 样条曲线从给定的点开始或终止，而且带有确定的切向量。但三次 B 样条曲线的始点及始点处的切向量是由 B 特征多边形的 P_0，P_1，P_2 三点决定的，一般来说不通过给定的始点。为了让 B 样条曲线从给定的点开始，并在该点具有给定的切向量，可以通过添加两个顶点的办法来解决。

假定 B 样条曲线的控制点序列 $P_i\,(i=0, 1, \ldots, n)$已经给定。如果要使曲线以 P_0 为起点且切于向量 P_0P_1，同时以 P_n 为终点且切于向量 $P_{n-1}P_n$，则只需在始端和终端各增加一个顶点 P_{-1} 及 P_{n+1}，使得向量 $P_{-1}P_0= P_0P_1$，$P_{n-1}P_n=P_nP_{n+1}$，这样在始端和终端所增加的 B 样条曲线段就可以满足上述要求。

由于考虑到 B 样条曲线在工程中应用的广泛性，下面对 B 样条曲线的性质作一简单的介绍（以三次 B 样条曲线为例）。

（1）端点性质及连续性

在连接处三次 B 样条曲线的一阶导数、二阶导数都连续，即三次 B 样条曲线有二阶导数的连续性。将此结论推广，即可得出 n 次 B 样条曲线 n–1 阶导数连续的论断。

（2）局部性

由式（33-4）可知，每一段三次 B 样条曲线由四个控制点的位置向量决定。因些，在三次 B 样条曲线中，改变一个控制点的位置，最多影响 4 个曲线段。所以，通过改变控制点的位置就可以对 B 样条曲线进行局部修改。这是一个非常重要的性质。

（3）扩展性

从式（33-4）可知，如果增加一个控制点，就相应地增加了一段 B 样条曲线。此时，原有的 B 样条曲线不受影响，而且新增的曲线段与原曲线的连接处具有一阶、二阶导数连续的特性。这一点是由 B 样条曲线本身的性质保证的，不需要附加任何条件。因而，对原有的 B 样条曲线加以扩展是很方便的。

33.2 B 样条曲线的生成

MATLAB 的样条曲线工具箱中提供的 B 样条曲线类函数如表 33-1 所示：

表 33-1　B 样条曲线类函数

函 数 名 称	简 单 介 绍
splst	显示生成 B 样条函数的 M 文件
spmak	生成 B 样条函数
spcrv	生成均匀划分的 B 样条函数
spapi	插值生成 B 样条函数
spap2	用最小二乘法拟合生成 B 样条函数
spaps	对生成的 B 样条曲线进行光滑处理
spcol	生成 B 样条函数的配置矩阵

1．splst 函数

用 splst 函数生成 B 样条函数的 M 文件，其调用格式为：

- splst。

2．spmak 函数

用 spmak 函数生成 B 样条函数，其调用格式为：

- spmak
- sp = spmak(knots, coefs)
- sp = spmak(knots, coefs, sizec)

spmak 生成 B 样条函数。输入参数 knots , coefs, sizec 分别表示节点矩阵，系数矩阵以及系数矩阵的大小。如果只输入“spmak”，则紧接着会出现提示，要求用户输入其他必须的参数(knots, coefs 等)。通过这种方式，实际存储的这种形式样条函数的数据很容易被修改，而不会对与之有关的各个 M 文件有影响。

通过对 coefs 参数设置为 d 维向量（如 2 维或 3 维等），可以控制 spmak 命令生成的是曲线还是曲面，

输入参数 knots 可以决定用 spmak 命令生成的 B 样条函数是一元函数还是多元函数。如果输入参数 knots 是节点序列，即形如$\{a_1, a_2, \ldots, a_m\}$的数组（数组 knots 必须是单调递增），则生成的 B 样条函数是一元函数，函数的阶次 k 等于数组 knots 的长度减去系数矩阵的列数，即 k=length(knots)-size(coefs,2)，这意味着系数矩阵的每一列元素都是 B 样条函数的系数。如果 B 样条函数是一元函数，则系数矩阵只有一列。特别地，若 B 样条函数是 d 元函数，d=size(coefs,1)，则系数矩阵只有一列。B 样条函数的取值区间为[knots(1) knots(end)]。

如果参数 breaks 是一个包含 m 个元素 $x_1 \ldots x_m$ 的数组，而每个元素又包含其他数组元素，即形如{{a1 a2 a3},{ a4 a6},{ a7 a9 a10 a11}}的数组，此数组的长度是 m，则 B 样条函数是 m 元函数。如果样条函数是矢量函数，则系数数组 coefs 的长度是（m+1）；如果样条函数是标量函数，则系数数组 coefs 的长度是 m，同时系数数组被下式代替，coefs = reshape(coefs, [1, size(coefs)])。此处，第 i 个输入向量的长度为：L=length(knots{i}-size(coef, i+1), i=1:m。同时，第 i 个输入向量的取值区间为[knots{i}(1) knots{i}(end)]。

【例 33-1】

```
>>spmak([1:6],[0:2])
ans =
    form: 'B-'
    knots: [1 2 3 4 5 6]
    coefs: [0 1 2]
    number: 3
    order: 3
dim: 1
```

上例表示在区间[1 6]内生成一条 B 样条曲线，它有 6 个节点元素和 3 个系数元素，函数的阶次 k=6-3=3，图形如图 33-1 所示。

又如，

```
>>spmak([1:3,4:6,7:9],[0:1,1:2],size([0:1,1:2]))
ans =
     form: 'B-'
     knots: [1 2 3 4 5 6 7 8 9]
     coefs: [0 1 1 2]
     number: 4
     order: 5
dim: 1
```

图形如 33-2 所示。

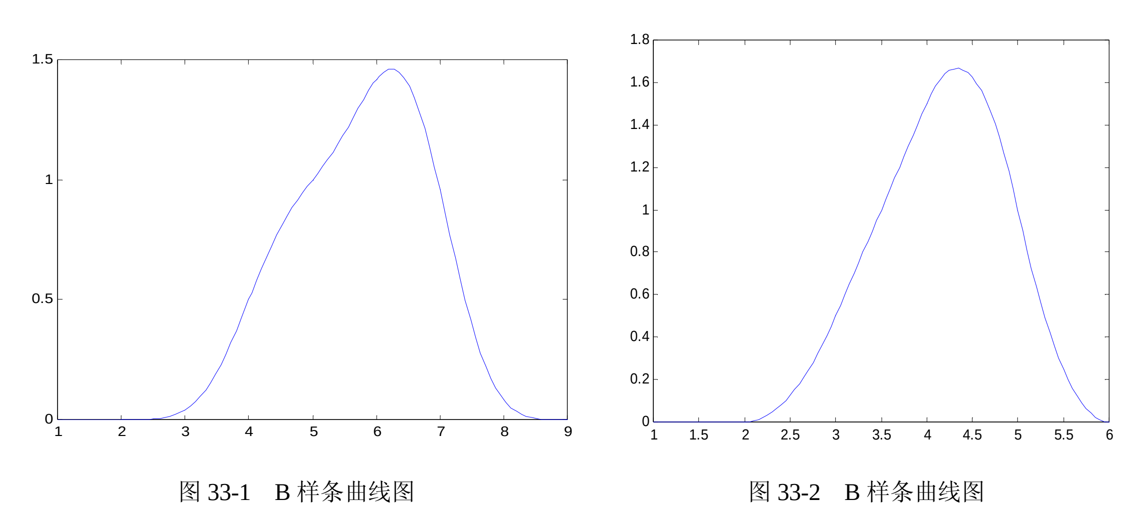


图 33-1　B 样条曲线图　　　图 33-2　B 样条曲线图

注意：如果节点数组 knots 不是单调递增或节点数组 knots 中的元素个数比系数数组 coefs 中少或者系数数组 coefs 为空，则会出错。

3．spcrv 函数

用 spcrv 函数生成均匀划分的 B 样条曲线，其调用格式为：

- spcrv(c)
- curve = spcrv(c, k, maxpnt)

spcrv(c, k, maxpnt)函数生成经过均匀划分的 B 样条函数 f, 其中输入参数 c 为 B 样条函数的系数矩阵，k 为函数的阶次，maxpnt 是允许划分的最大段数，曲线的形式如下：

$$f: t \mapsto \sum_{j=1}^{n} B(t-k/2 \mid j,\dots,j+k) * c(j), \frac{k}{2} \leqslant t \leqslant n+\frac{k}{2}$$

式中，$B(.|a,\dots,z)$是在节点 $a,\dots,z$ 的 B 样条函数，n 为函数中系数的个数，即[d, n]=size(c)，样条函数的阶次 k 默认为 4。

【例 33-2】 下面所示生成一条经过离散的点，并光滑处理后的 B 样条曲线。

```
>>points = [0 0 1 1 0 -1 -1 0 0 ;0 0 0 1 2 1 0 -1 -2];
plot(points(1,:),points(2,:),':')
values = spcrv(points,3);
hold on, plot(values(1,:),values(2,:)), hold off
```

图形如图 33-3 所示。

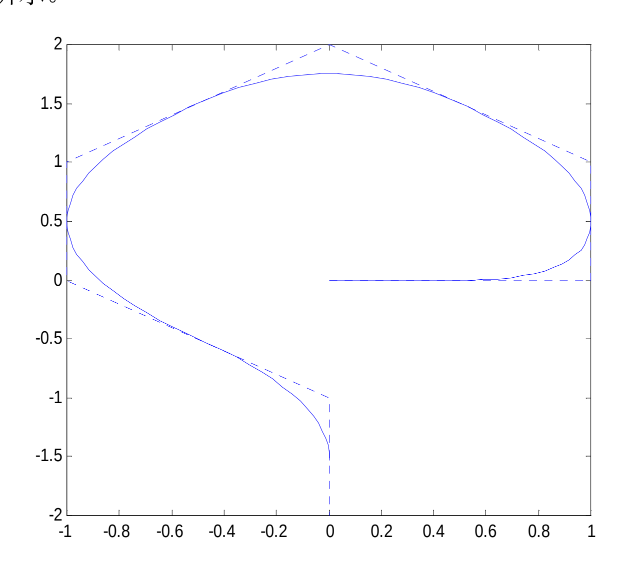


图 33-3　光滑处理的 B 样条曲线图

4．spapi 函数

用 spapi 函数插值生成 B 样条函数，其调用格式为：

- spline = spapi(knots,x,y)
- spline = spapi(k,x,y)

给定节点数组 knots 与内插数组 x，y，返回插值生成的 B 样条函数 f。函数的阶次为 k=length(knots) - length(x)。其中，数组 x, y 满足：

$$y(:,i) = f(x(i)), \text{all } i$$

它包括了数组 x 有重复值的情况，即 $D^{m(i)}f(x(i)) = y(:,i)$。注意，数组 x 必须单调递增。式中，m = knt2mlt(x)，即 $m(i) := \#\{j < i: x(j) = x(i)\}$，式中的 y 表示对 B 样条函数 f 求 m-1 次导数后在 x 处的导数值。

如果代替输入节点数组 knots，而输入样条函数期望的阶次 k，则要用操作器类函数 aptknt 从数组 x 中求出一个可用的（尽管可能不是最优的）节点数组 knots。

【例 33-3】

>>spapi([0 0 0 0 1 2 2 2 2], [0 1 1 1 2], [2 0 1 2 -1])

上例表示在区间[0 2]内生成一条 B 样条曲线，但此曲线要通过给定的内插节点，即当节

点为 1 时，要满足 $f(0+)=2, f(1)=0, Df(1)=1, D^2f(1)=2, f(2-)=-1$。

结果如图 33-4 所示。

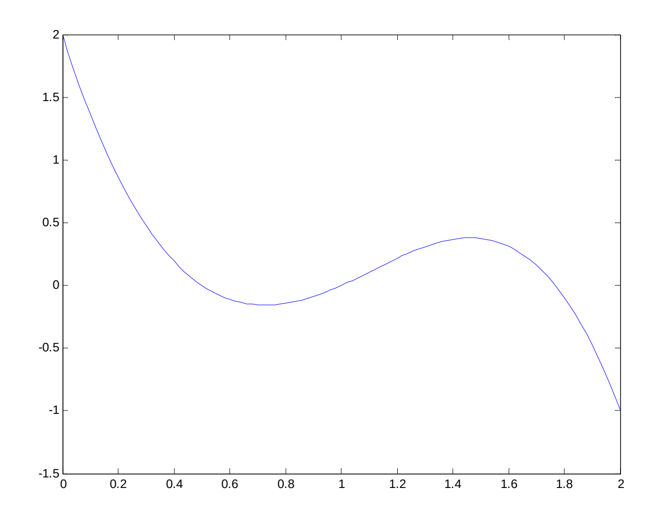


图 33-4　一元函数插值生成的 B 样条曲线图

下面是一个二元函数内插生成 B 样条曲面的例子。

```
>>x = -2:.5:2;
y=-1:.25:1;
[xx, yy] = ndgrid(x,y);
z = exp(-(xx.^2+yy.^2));
sp = spapi({3,4},{x,y},z);
fnplt(sp)
```

图形如图 33-5 所示。

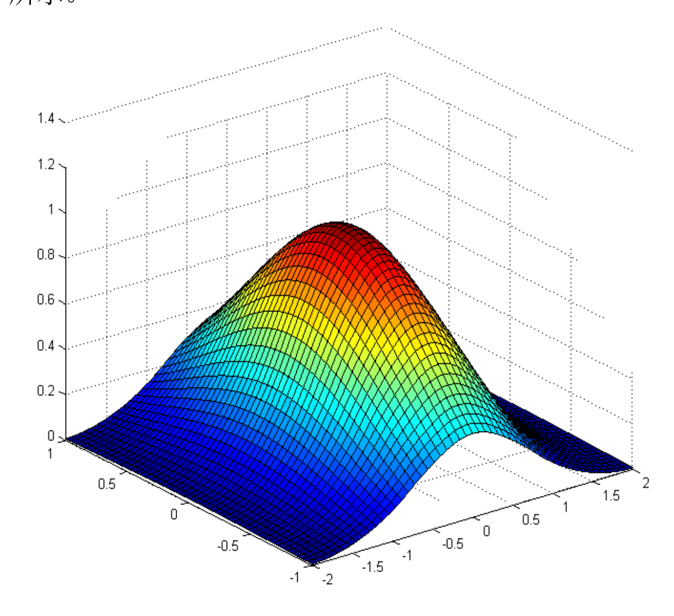


图 33-5　二元函数插值生成 B 样条曲面图

注意：输入参数中的节点数组 knots 与数组 x 在插值时必须满足 Schoenberg-Whiteney 条

件。例如，如果数组 *x* 单调递增，则对所有的 *j* 必须满足 knots(j) < x(j) < knots(j+k)。这里节点 knots(1)与节点 knots(n+k)可能相等。在多变量情况下，每个变量都必须满足上述条件。

在单变量情况下，如果数组 *x* 不是单调递增，则必须对数组 *x* 与 *y* 重新排列，以使它们符合单调递增。在多变量情况下，每个变量都必须如此重新调整。

5．spap2 函数

用 spap2 函数，使用最小二乘法拟合生成 B 样条曲线，其调用格式为：

- sp = spap2(knots,k,x,y)
- sp = spap2(l,k,x,y)
- sp = spap2(knots,k,x,y,w)

该函数生成对所有的 *j* 满足 *y*(:, *j*)=*f*(*x*(*j*)), 阶次为 *k* 的 B 样条函数。其中，sp = spap2(knots, k, x, y, w)中的 knots 是节点数组，w 为权重，默认为 1。如果数组 x 满足 Schoenberg-Whiteney 条件，即 knots(j) < x(j) < knots(j+k), j=1,…,length(x)，此处的 length(x)=length(knots)-k，则生成一条单调递增的 B 样条曲线。因为选择合适的节点数组 knots 可能很难，所以可以代替输入节点数组 knots，而输入分段多项式的段数 l。此时 spap2 函数能自动找到一个合适的节点数组 knots。

如果输入参数中的 knots 是 m 维的向量，则输入参数 x 也必须是 m 维的向量。此时，如果有输入参数 w、k，则 w 与 k 也必须是 m 维的向量。同时，如果样条函数是矢量形式，则 y 必须是长度为 m+1 的向量。但是，如果样条函数是标量形式，则 y 为单变量形式的数组，其长度为 m。对所有的 i1, ..., im，y(:,i1,...,im) 是与向量 [x{1}(i1),...,x{m}(im)]相对应的，用于样条拟合的 Y 向坐标值。

【例 33-4】 下面的例子用最小二乘法拟合数组 x、y 生成 B 样条曲线，在两个端点 0 与 5，以及断点 3 处求二阶连续偏导数。

```
>>x=linspace(0,2*pi,51);
y=cos(x)+0.2*(rand(size(x))-0.5);
fnplt(spap2(augknt([0 3 5],4),4,x,y))
```

结果如图 33-6 所示。

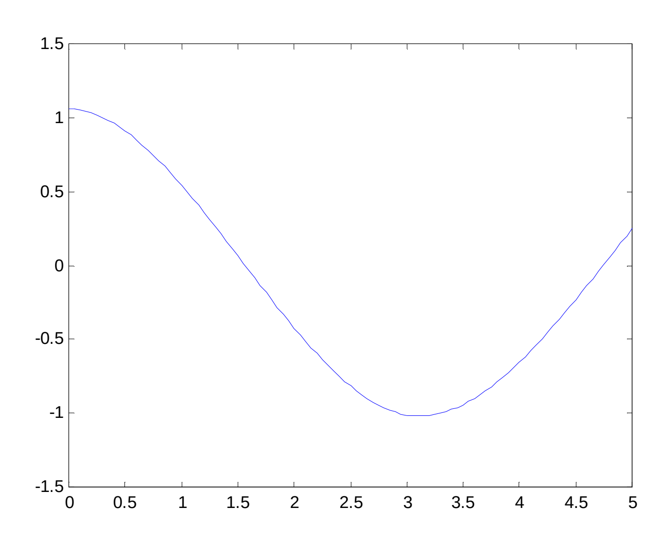


图 33-6 最小二乘法拟合生成的 B 样条曲线图

下例是输入分段多项式的段数生成样条曲线的例子。

```
>>x = -2:.5:2;
y=cos(x);
w = ones(size(x));
w([1 end]) = 100;
pp=spap2([1 3 6],2,x,y,w);
fnplt(pp)
```

结果如图 33-7 所示。

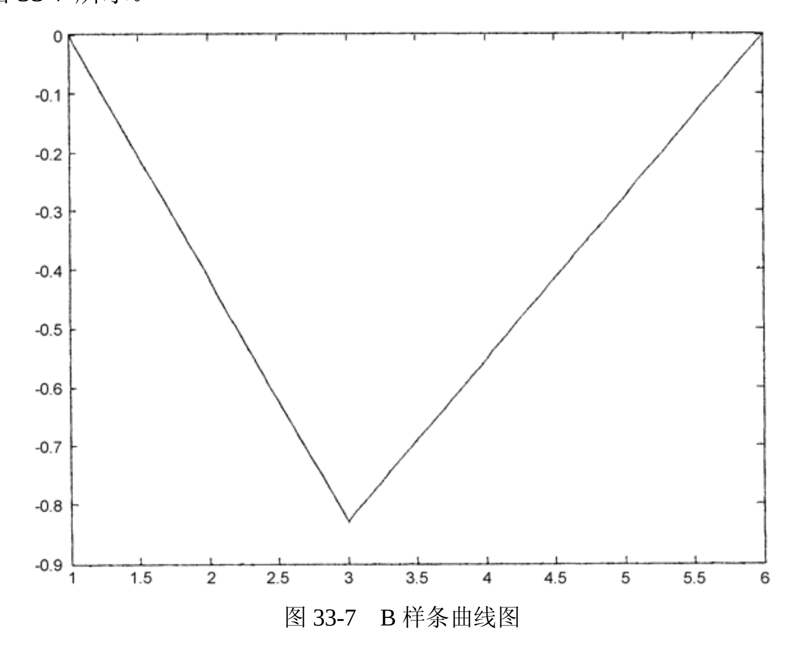


图 33-7　B 样条曲线图

6．spaps 函数

用 spaps 函数对生成的 B 样条曲线进行光滑处理，其调用格式为：

- sp = spaps(x,y,tol)
- [sp, values] = spaps(x,y,tol,arg1,arg2)

输入参数中的数组 x, y 表示用于生成 B 样条函数的坐标值，tol 表示光滑时的允许精度。该函数表示在给定数据下，在容许的精度范围内生成最光滑的 B 样条曲线。在这里，最光滑意思是说下式的值最小。

$$f(D^m f) = \int_{x(1)}^{x(n)} \left(D^m f\right)^2 \qquad （33\text{-}7）$$

进一步讲，就是根据数组 x 计算出的 B 样条函数值与给定的数组 y 之间的距离最小，即：

$$E(f) = \sum_{j=1}^{n} w(j)\left(y(j) - f\left(x(j)\right)\right)^2 \qquad （3\text{-}8）$$

最小。

在式（33-7）中，m 的默认值是 2，即平滑生成三次 B 样条曲线。通过设置输入参数中的一个 argi 为 1 或 3，使 m=1 或 3，表示平滑生成一次或五次 B 样条曲线。权向量 w 的默认值使函数 $E(f)$与下式的距离最小。

$$\int_{x(1)}^{x(n)}\left(y-f\right)^2$$

可以通过设置其中的一个可选参数 argi 来提供权向量值。此时，要求该 argi 数组与 x 数组的长度相等，且该 argi 数组中的元素必须为正数。

拟合用的扭结点（节点）数组可以是 d 维向量形式。这时 y 数组的大小是[d,n]。且式（33-7）与（33-8）为各维元素的总和。也就是说，如果 $f(x)$是一个 d 维的函数数组{$f1(x)$,..,$fd(x)$}，则 $E(f)=E(f1)+\ldots+E(fd)$。

如果 x 是一个长度为 r 的数组，则 y 是一个与 x 长度相等的数组。如果函数是标量形式，则数组 y 的大小为[length(x{1}),...,length(x{r})]。如果函数是向量形式，则数组 y 的大小为[d,length(x{1}),...,length(x{r})]。另外，可选参数 m（通过输入参数 arg1 或 arg2 设定）必须为 1, 2 或 3。同时，可选参数 w（也通过输入参数 arg1 或 arg2 设定）必须是一个长度为 r 的数组。w{i}为空，则使用默认值。w{i}是一个长度与 x{i}相等的正向量。

【例 33-5】

```
>>x=linspace(0,2*pi,51);
noisy_y=cos(x)+0.2*(rand(size(x))-0.5);
fnplt(spaps(x,noisy_y, 1.e-2,3),'r',2)
```

上例表示平滑生成五次 B 样条曲线，平滑精度值为 1.e-2。结果如图 33-8 所示。

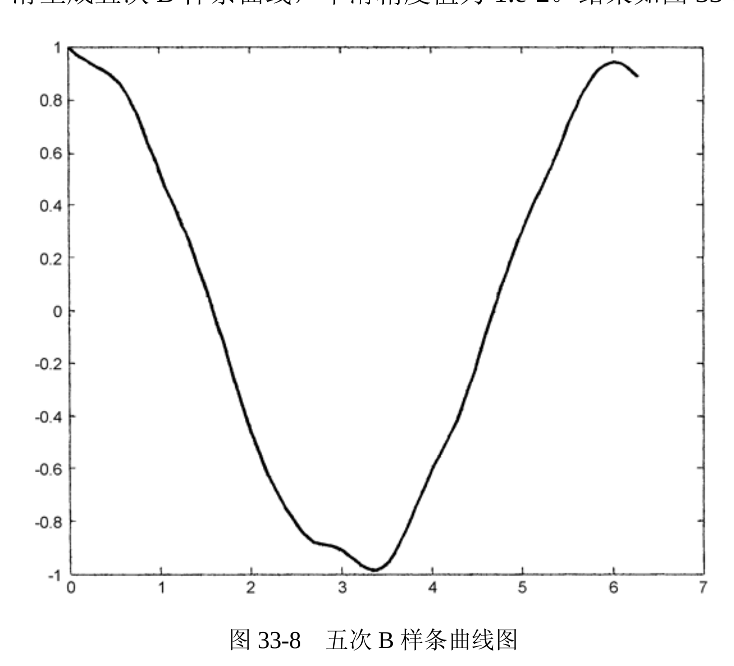


图 33-8　五次 B 样条曲线图

注意：如果数组 x 中的元素不是单调递增，则必须对数组 x、y 进行重新排列，以使它们符合单调递增。

7．spcol 函数

用 spcol 函数生成 B 样条曲线的配置矩阵，其调用格式为：

- spcol(knots, k, tau)
- colloc = spcol(knots, k, tau, arg1, arg2)

此函数通过下式生成矩阵：

$$colloc := (D^{m(i)}Bj(tau(i)))$$

上式中，Bj 为第 j 条 B 样条曲线的阶次，tau 是一个单调递增的节点数组，m= knt2mlt(tau), 即 m(i) := #{j < i:tau(j) = tau(i)}

如果可选参数 arg1(或 arg2)是一个前两个字母与函数 slvblk 的前两个字母相同的字符串，则返回的是块矩阵。该矩阵用 slvblk 函数或 bkbrk 才能读出。

如果可选参数 arg1（或 arg2）是一个前两个字母与函数 sparse 的前两个字母相同的字符串，则返回的矩阵是稀疏矩阵。

如果可选参数 arg1（或 arg2）是一个前两个字母与函数 noderiv 的前两个字母相同的字符串，则对所有的 i，m(i)=1。

【例 33-6】

```
>>spcol([1:6],3,1+[2:4])
ans =
    0.5000   0.5000        0
         0   0.5000        0.5000
         0        0        0.5000
```

上述命令生成阶次为 3，节点数组为[1: 6]的 B 样条曲线的配置矩阵，矩阵各行的元素分别是在节点 2.1, 3.1, 4.1 处的函数值。第一条曲线通过节点 1, 2, 3, 4 生成 B 样条曲线，它的值存储在矩阵的第 1 行中。特别地，第 1 行中的最后一个元素值为 0，因为它是在节点 4.1 处的 B 样条函数值，这是最后一个节点 4 右边的节点，它已超出了节点的范围。

如果设可选参数 arg1(或 arg2)为字符串"sl"，则输出函数 bkbrk 可读出的矩阵，如下所示：

```
>>bkbrk(spcol([1:6],3,1+[2:4],'sl'))
block 1 has 1 row(s)
    0.5000   0.5000        0
next block is shifted over 1 column(s)

block 2 has 1 row(s)
    0.5000   0.5000
next block is shifted over 1 column(s)
block 3 has 1 row(s)
    0.5000
next block is shifted over 1 column(s)
```

上述结果即为：

```
0.5000   0.5000   0
0        0.5000   0.5000
0        0        0.5000
```

第 34 章　有理样条曲线

34.1　基本原理

为了更方便地控制曲线的形状，基于齐次坐标的概念，产生了用有理参数多项式构造的样条曲线，即有理样条曲线。采用有理参数多项式有两个重要优点。第一，有理参数多项式具有几何和透视投影变换不变性。例如，要产生一条经过透视投影变换的空间曲线，对于用无理多项式表示的曲线，第 1 步需生成曲线的离散点，第 2 步再对这些离散点作透视变换，得到要求的曲线。对于用有理多项式表示的曲线，第 1 步对定义曲线的控制点作透视投影变换，第 2 步用变换后的控制点生成要求的曲线。显然，后者比前者的工作量小许多。第二，用有理参数多项式可精确地表示圆锥曲线、二次曲面等，进而可统一几何造型算法。

已知 $\boldsymbol{s}(x)$、$w(x)$是两个样条函数，其中 w 是标量函数，$\boldsymbol{s}$ 是矢量函数，则

$$\boldsymbol{r}(x) = \boldsymbol{s}(x)/w(x)$$

使 $\boldsymbol{s}(x)$变成有理样条函数 $\boldsymbol{r}(x)$。注意，在 x 处的函数值 $w(x)$一般不为零。

有理样条函数的应用非常广泛。因为与一般的样条函数相比，用有理样条函数可精确地表示圆、圆锥等基本的形状曲线。例如，利用 MATLAB 样条工具箱中的函数 rsmak，

```
>>circle = rsmak('circle');
fnplt(circle), axis square
```

即可生成半径为 1 的圆，如图 34-1 所示。

使用 MATLAB 样条工具箱中的其他函数，很容易把圆变成其他的形状。例如，使用下述操作就可以把圆变成倾斜 45°、拉长的椭圆。

```
>>circle = rsmak('circle');
fnplt(circle), axis square;
ellipse = fncmb(circle,[2 0;0 1]);
s45 = 1/sqrt(2);
rtellipse = fncmb(fncmb(ellipse, [s45 -s45;s45 s45]), [1;1] );
hold on, fnplt(rtellipse), hold off
```

把椭圆与圆画在一起，如图 34-2 所示。

下面是一个在三维空间中生成有理样条曲面的例子：

```
>>southcap=rsmak('southcap');
fnplt(southcap),axis square
```

生成的图形如 34-3 所示。

对上述图形经过简单调整，就可以获得一个球体。例如，下面的操作将生成 2/3 的球体。

```
>>southcap = rsmak('southcap');
fnplt(southcap)
xpcap = fncmb(southcap,[0 0 -1;0 1 0;1 0 0]);
ypcap = fncmb(xpcap,[0 -1 0; 1 0 0; 0 0 1]);
northcap = fncmb(southcap,-1);
```

```
hold on, fnplt(xpcap), fnplt(ypcap), fnplt(northcap)
axis equal, shading interp, view(-115,10), axis off, hold off
```

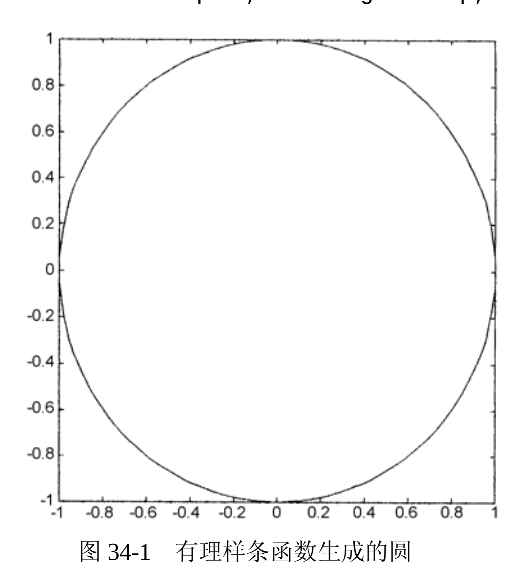


图 34-1　有理样条函数生成的圆

图 34-2　有理样条函数生成的圆与椭圆

生成的图形如 34-4 所示。

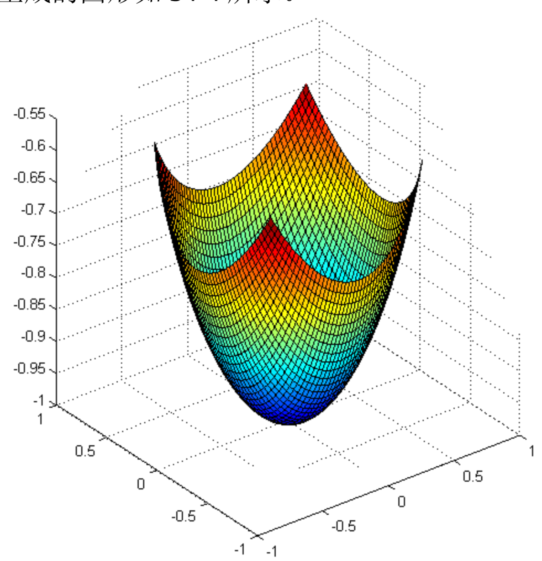


图 34-3　有理样条函数生成的曲面



图 34-4　有理样条函数生成的 2/3 球体

在有理样条函数 $r(x) = \mathbf{s}(x)/w(x)$中的两个函数 $\mathbf{s}(x)$, $w(x)$不一定相关。它们甚至可以是形式完全不同的样条函数。但是在 MATLAB 样条工具箱中，要求它们是同一种形式的样条函数，甚至要求两函数的阶次也相同，并且要有同样的节点数组。在上述要求下，可以使用下式来代表有理样条函数：

$$R(x)=[s(x);w(x)]$$

从 $R(x)$很容易获得 $r(x)$。例如，如果 v 是函数 R 在 x 处的值，即 $v=R(x)$，则 $r(x)=v(1:\text{end}-1)/v(\text{end})$。进一步，如果 $\text{d}v=DR(x)$，则 $Dr(x)=(\text{d}v(1:\text{end}-1)-\text{d}v(\text{end})*v(1:\text{end}-1))/v(\text{end})$。一般地，由 Leibniz 公式：

$$D^j s = D^j (wr) = \sum_{i=0}^{j} \binom{j}{i} D^i w D^{j-i} r$$

因此，

$$D^j r = \left[D^j s - \sum_{i=1}^{j} \binom{j}{i} D^i w D^{j-i} r \right] / w$$

由此，我们可以计算有理样条函数 $r(x)$（即 $R(X)$）的导函数。

使用样条工具箱中提供的对样条函数进行操作的函数（操作器类函数），可以很容易地对有理样条函数进行操作。但是，积分函数 fnint 不能用，因为有理样条函数进行积分后不一定是有理样条函数。同样，微分函数 fnder 与 fndir 也不能用。操作器类中的 fntlr 函数可以计算有理样条函数在给定点处的各阶偏导数值。

一种特殊形式的有理样条函数是 NURBS，即非均匀有理 B 样条函数。此时，样条函数 $s(x)$、$w(x)$都是 B 样条函数，即

$$s = \sum_i B_i v(i) a(:,i) \qquad w = \sum_i B_i v(i)$$

前面的用有理样条函数 rsmak 生成的椭圆也可以用下述方法来实现：

```
>>x = [1 1 0 -1 -1 -1  0  1 1];
y = [0 1 1  1  0 -1 -1 -1 0];
s45 = 1/sqrt(2);
w =[1 s45 1 s45 1 s45 1 s45 1];
circle = rsmak(augknt(0:4,3,2), [w.*x;w.*y;w]);
fnplt(circle)
```

结果如图 34-5 所示。

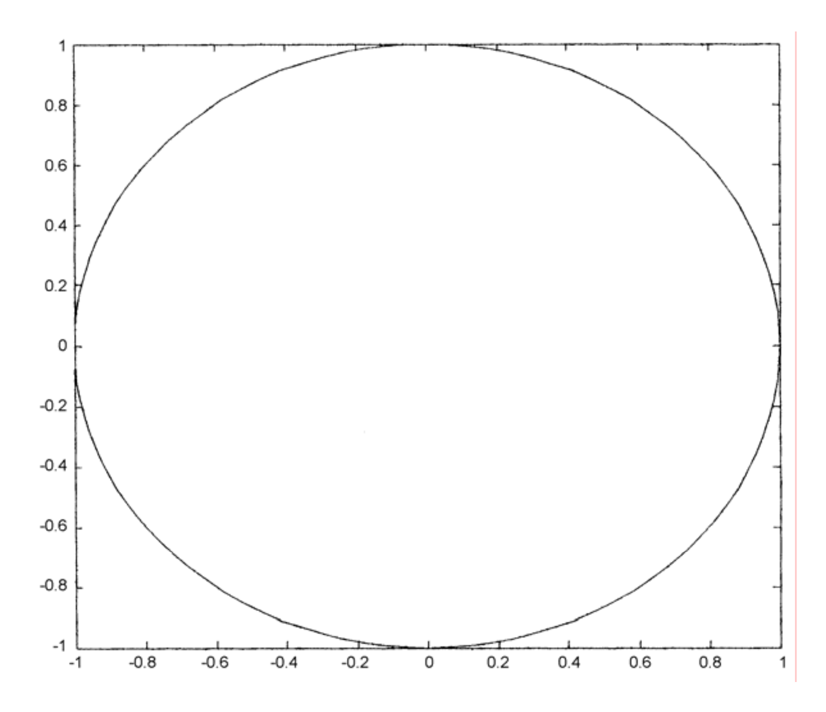


图 34-5　有理样条函数生成的椭圆

34.2　有理样条函数的生成

MATLAB 样条工具箱中提供的生成有理样条曲线的函数主要有 rpmak 与 rsmak 两个。它们的功能与语法基本相同，现一并介绍如下。

利用 rpmak 与 rsmak 这两个函数生成有理样条函数，其调用格式为：

- rp = rpmak(breaks, coefs)
- rp = rpmak(breaks, coefs, d)
- rs = rsmak(knots, coefs)
- rs = rsmak(shape, parameters)

在给定断点（或节点）breaks（knots）与系数 coefs 下生成有理样条曲线。输入参数 d 是样条函数的维数。与函数 rpmak 相比，函数 rsmak 还可生成标准的几何形状曲线，如圆锥等。

除了用函数 rpmak(breaks,coefs)生成的函数被标明为有理样条函数外，在其他方面，rpmak(breaks,coefs)函数与 ppmak(breaks, coefs)函数很相似。为了说明上述意思，假设 R 为用 ppmak(breaks, coefs)生成的分段多项式样条函数，$r(x)=s(x)/w(x)$是用 rpmak(breaks,coefs)生成的有理样条函数。假设 $V=R(x)$，则 $r(x)=v(1:\text{end}-1)/v(\text{end})$，即 $R(x)=[s(x);w(x)]$。与此相对应，r 的维数比 R 的维数低一维。特别地，R 的维数至少 2 维，即系数 coefs 必须是一个维数大于 1（即 $d>1$）的向量。对于输入参数 breaks 与 coefs 的详细说明请参考函数 ppmak 的描述。

函数 spmak(knots,coefs)与 rsmak(knots,coefs)也很相似。rsmak(knots ,coefs)生成的是 B 样条形式的有理样条函数。对于输入参数 knots 与 coefs 的详细说明请参考函数 spmak 的描述。

通过设置输入参数 shape 及可选参数 parameters，rsmak(shape,parameters)函数可以生成圆锥等标准形式的有理样条函数。可选的参数有：

rsmak(‘circle', radius, center)
rsmak(‘cone', radius, halfheight)
rsmak(‘cylinder', radius, height)
rsmak(‘southcap', radius, center)

通过投影变换，上述形状的样条曲线可转变成其他形状的样条曲线，如椭圆等。在操作器类函数中，除了函数 fnint, fnder, fndir 外，其他形如 fn…的函数都可以用来进行仿射变换。

【例 34-1】

runges = rsmak([-5 -5 -5 5 5 5],[1 1 1; 26 -24 26]);

与

rungep = rpmak([-5 5],[0 0 1; 1 -10 26],1);

都在区间[-5 5]内生成有理样条函数 $r(x) = 1/(x^2 + 1)$。但是在[-5 5]区间外，函数 rsmak 生成的函数值等于零，而函数 rpmak 生成的函数值等于 $r(x) = 1/(x^2 + 1)$生成的函数值（此时 x 在区间[-5 5]外）。上述命令生成的图形如 34-6 所示。

【例 34-2】　下面生成一个旋转的圆锥。

```
>>fnplt(fncmb(rsmak('cone',1,2),[0 0 -1;0 1 0;1 0 0]))
axis equal, axis off, shading interp
```

生成的图形如 34-7 所示。

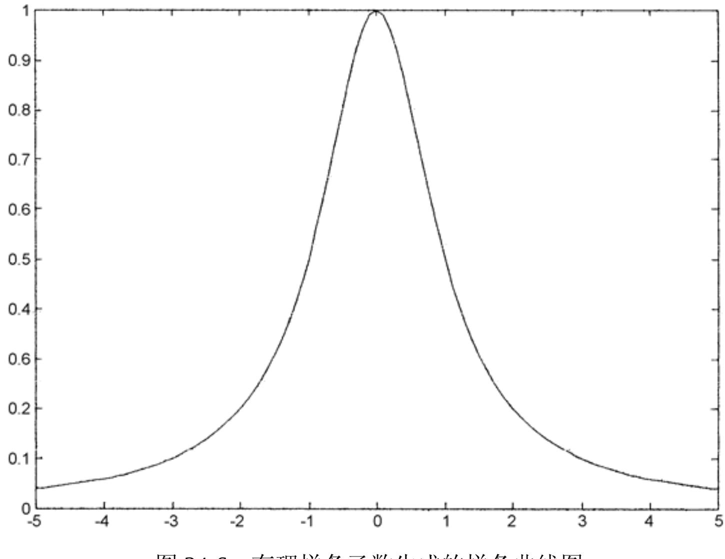


图 34-6　有理样条函数生成的样条曲线图

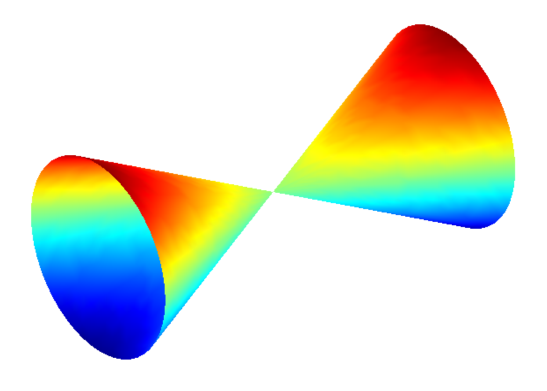


图 34-7　有理样条函数生成的旋转圆锥

第 35 章　操作器类函数

MATLAB 样条工具箱中提供了一些专门针对样条函数进行操作的函数（操作器类函数）、样条曲线端点与节点处理类函数及方程求解类函数。它们中的一些既可以对分段多项式样条函数进行操作，又可以对 B 样条函数以及有理样条函数进行操作；还有一些可以对节点进行操作。下面，先对操作器类函数做一介绍，其他的两类函数在下面几节介绍。MATLAB 的样条工具箱中提供的操作器类函数如表 35-1 所示。

表 35-1　样条函数的操作器类函数

函 数 名 称	简 单 介 绍
fnval	计算在给定点处的样条函数值
fmbrk	返回样条函数的某一部分（如断点或系数等）
fncmb	对样条函数进行算术运算
fn2fm	把一种形式的样条函数转化成另一种形式的样条函数
fnder	求样条函数的微分(即求导数)
fndir	求样条函数的方向导数
fnint	求样条函数的积分
fnjmp	在间断点处求函数值
fnplt	画样条曲线图
fnrfn	在样条曲线中插入断点
fntlr	生成 Tarylor 系数或 Taylor 多项式

下面，按上表所列，对每个函数逐一进行介绍。

1．fnval 函数

该函数计算给定点处的样条函数值，其调用格式为：

- values = fnval(f,x)
- values = fnval(x,f)
- values = 　fnval(f,x,'l')

fnval(f, x), fnval(x, f)计算函数 f 在给定点 X 处的函数值，返回的是 $f(x)$的矩阵形式，输出结果主要看函数 f 是一元函数还是多元函数。

如果 f 是一元函数，则输出结果是一个大小为[d*m,n]的矩阵，其中 d 是函数 f 的最高次幂（函数的阶次），$[m,n]$是矩阵 x 的大小。如果函数 f 在点 x 处有间断，则返回的是 $f(x+)$，即右连续的函数值。但是，如果 x 等于取值区间的右端点值时，则返回的是 $f(x–)$，即左连续的函数值。

如果可选参数'l'也给出，则函数是左连续。也就是说，如果函数 f 在 x 处有间断，则返回的函数值是 $f(x–)$，即左连续的函数值。除非，x 等于取值区间的左端点值，此时返回的是 $f(x+)$，即右连续的函数值。

如果 f 是多元 d 次函数，并且 f 在整个自变量的取值区间上是连续的，则输出结果如下所示：

$$\text{values}=\begin{cases}[d*m,n], & 当x的大小为[m,n]时\\ [d,n_1,n_2,...,n_m], & 当d>1且x的大小为[n_1,n_2,...,n_m]时\\ [n_1,n_2,...,n_m], & 当d=1且x的大小为[n_1,n_2,...,n_m]时\end{cases}$$

此时，fnval(csapi(x,y),xx)的返回结果与 csapi(x,y,xx）相同。

【例 35-1】

```
>>x=2*pi*[0 1 0.1 0.2 0.9];
y=cos(x);
cs=csapi(x,y);    % 插值生成三次样条函数
fnval(cs,[0 2*pi])
ans =
   1   1
```

2．fnbrk 函数

该函数返回样条函数某一部分（如节点或系数等）的信息，其调用格式为：

- out = fnbrk(f,part)
- pp = fnbrk (pp,[a b])
- pp = fnbrk(pp,j)
- fnbrk(f)

out = fnbrk(fn,part)返回样条函数 fn 中被 part 标明的那一部分，输出参数 out 可作为函数 spmak 或 ppmak 的输入参数。特别地，out=fnbrk(fn,‘from’)返回一个字符串，表明样条函数 fn 的类型。

如果 fn 是 B 样条函数，则 part 可取：‘knot’或 ‘t’，‘coefs’，‘number’，‘order’，‘dimension’和‘interval’；此时分别返回：B 样条函数的节点数组、B 样条函数的系数数组、B 样条函数的系数个数、B 样条函数的阶、B 样条函数系数的维数、自变量的取值区间。

如果 fn 是分段多项式样条函数，则 part 可取：‘breaks’，‘coefs’，‘piece’或‘l’，‘order’，‘dimension’和‘interval’；此时分别返回：分段多项式函数的端点数组、分段多项式函数的系数数组、分段多项式函数的段数、分段多项式函数的阶、分段多项式函数系数的维数、自变量的取值区间。

另外，part 也可以是某个区间，如[a b]，此时，输出参数 pp 是该区间范围内的分段多项式函数。除此之外，part 也可以取正整数，如 j，此时，输出参数 pp 是第 j 段分段多项式函数。

如果函数 fn 是多元函数，则返回的是与之相对应的多元部分。也就是说，返回的节点与端点是矩阵形式，返回的分段多项式系数向量一般大于等于二维，返回的分段多项式系数的维数、系数的个数以及分段数是向量形式。

如果没有标明输出参数，即调用格式为 fnbrk(f)，只有一个输入参数，此时样条函数的各个部分的信息输出到屏幕上。

【例 35-2】

```
>>x=2*pi*[0 1 0.1 0.2 0.9];
y=cos(x);
cs=csapi(x,y);
out=fnbrk(cs,'breaks')
out =
        0    0.6283    1.2566    5.6549    6.2832
```

【例 35-3】

```
>>x=2*pi*[0 1 0.1 0.2 0.9];
```

```
y=cos(x);
cs=csapi(x,y);
out=fnbrk(cs,'coefs')
out =
    0.3830   -1.1133    0.2443    1.0000
    0.3830   -0.3914   -0.7011    0.8090
   -0.0311    0.3305   -0.7393    0.3090
   -0.0311   -0.0792    0.3660    0.8090
```

【例 35-4】

```
>>x=2*pi*[0 1 0.1 0.2 0.9];
y=cos(x);
cs=csapi(x,y);
out=fnbrk(cs,'piece')
out =
    4
```

【例 35-5】

```
>>x=2*pi*[0 1 0.1 0.2 0.9];
y=cos(x);
cs=csapi(x,y);
out=fnbrk(cs,'order')
out =
    4
```

【例 35-6】

```
>>x=2*pi*[0 1 0.1 0.2 0.9];
y=cos(x);
cs=csapi(x,y);
out=fnbrk(cs,'dim')
out =
    1
```

【例 35-7】

```
>>x=2*pi*[0 1 0.1 0.2 0.9];
y=cos(x);
cs=csapi(x,y);
out=fnbrk(cs,'interval')
out =
    0    6.2832
```

3．fncmb 函数

该函数对样条函数进行算术运算，其调用格式为：

- fn = fncmb(function, matrix)
- fn = fncmb(function, function)
- fn = fncmb(function, matrix, function)
- fn = fncmb(function, matrix, function, matrix)
- fn = fncmb(function, 'op', function）

这个函数对样条函数进行比例变换以及加、减、乘、除等标准的四则运算。另外，也可

以用一个矩阵与函数进行乘法操作。甚至，还可以对两个不同形式的一元样条函数逐点进行相加或相乘操作。其中，

fn = fncmb(function, matrix) 使函数与矩阵或标量相乘。

fn = fncmb(function, function) 使两个相同形式的函数相加。

fn = fncmb(function, matrix, function) 使第 1 个函数与矩阵或标量相乘，然后与相同形式的第 2 个函数相加。

fn = fncmb(function,matrix,function,matrix) 使两个相同形式的函数按各自矩阵或标量所给的权重进行线性组合。

fn = fncmb(function, 'op', function)使分段多项式样条函数相加（op=+）、相减(op=-)或两个不同形式的函数逐点相乘(op=*)。

例如，

fncmb(fn, 3.5)返回函数 fn 的系数乘以 3.5 的结果。

fncmb(f, g) 返回 f+g 的结果。

fncmb(f, 3, g, -4)返回的是 3*f-4*g 的结果。

fncmb(f, 3, g)返回 3*f+g 的结果。

设 f 是一个 f(x)形式的函数，则 f3=fncmb(f, [1; 2; 3]) 返回的是(f(x), 2f(x), 3f(x))。

设函数 f 的样条曲线是三维空间中的一个面，即 f 是一个二元三次函数，则 f2=fncmb(f,[1 0 0; 0 0 1]) 返回面在(*x*,*z*)平面上的投影。

设 *t* 是有 *n*+*k* 个控制点的节点数组，*a* 是一个 *n* 列的矩阵，则 fncmb(spmak(t,eye(n,n),a) 等同于 spmak(t, a)。

【例 35-8】

```
>>x=0:4; y=-2:2; s2=1/sqrt(2);
clear v
v(3,:,:)=[0 1 s2 0 -s2 -1 0].'*[1 1 1 1 1];
v(2,:,:)=[1 0 s2 1 s2 0 -1].'*[0 1 0 -1 0];
v(1,:,:)=[1 0 s2 1 s2 0 -1].'*[1 0 -1 0 1];
sph = csape({x,y},v,{'clamped','periodic'});
fnplt(fncmb(sph,[1 0 0;0 1 0]))
```

上述命令返回的样条函数 sph 在(x, y)平面上的投影，结果如图 35-1 所示。

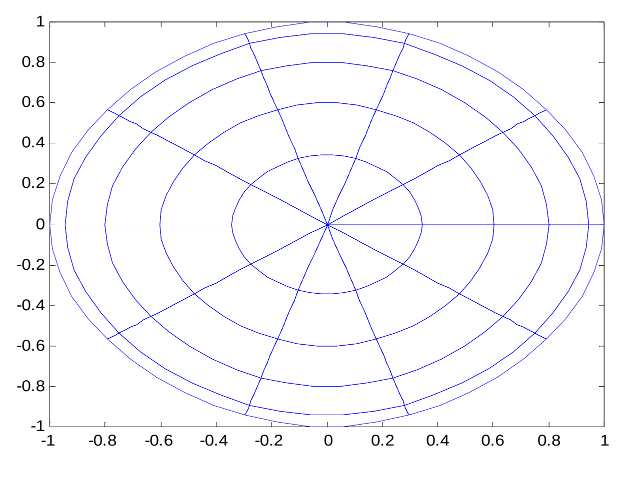


图 35-1　函数投影图

注意：在上面各种调用格式中，除了 fn = fncmb(function,matrix)外，其他的调用格式都要求函数 function 必须是一元函数。

另外，如果输入参数中有两个 function，则必须要求它们是同一类型的样条函数，而且要有相同的节点或端点、相同的函数阶、相同的维数，只有 fn = fncmb(function,'op',function)例外。

MATLAB 并不对上述匹配条件进行显式的检验，但如果两个系数行列式大小不一致，则 MATLAB 会给出一个大小不匹配的出错消息。

4．fn2fm 函数

利用该函数把一种形式的样条函数转化成另一种形式的样条函数，其调用格式为：

- g = fn2fm(f, form)
- sp = fn2fm(pp, 'B-', sconds)

参数中的 f 与 g 都是样条函数，只是函数的形式不同，输入参数 form 可以取以下值：'B-'(或'sp')（B 样条函数），'pp'（分段多项式样条函数），'BB'（BB 样条函数）。其中，BB 样条函数是节点数组中的每个元素都是 k 重次的一种特殊类型的 B 样条函数。

另外，为了与以后或当前的 MatLab 版本的 ppval 函数相兼容，输入参数中的 form 也可以取'MA'。此时，函数 *f* 是一元函数，返回函数 *g* 是一个分段多项式样条函数，但它能被现在版本的 ppval 函数兼容。

如果参数 form 是'B-'，而 *f* 是分段多项式样条函数，则内部每个断点（节点）处 *f* 函数的平滑性要推算得到。对于内部每一个断点，断点的一阶导数是连续的。默认精度是 1.e–12，用户可以通过给定参数 sconds 来设置精度值。参数 sconds 的取值在区间[0 1]内。另外，用户也可以通过输入参数 sconds(i)对内部不同的断点设置不同的平滑精度，此时，必须对内部每一个断点都设置平滑精度。

【例 35-9】

```
>>x=2*pi*[0 1 0.1 0.2 0.9];
y=cos(x);
cs=csapi(x,y);
fn2fm(cs,'B-')
ans =
     form: 'B-'
     knots: [0 0 0 0 1.2566 6.2832 6.2832 6.2832 6.2832]
     coefs: [1.0000 1.1023 -1.3159 0.6152 1]
     number: 5
     order: 4
     dim: 1
```

而，

```
>>fn2fm(cs,'MA')
ans =
  Columns 1 through 11
  30.0000   1.0000   4.0000   0      0.6283    1.2566   5.6549
   6.2832   4.0000   0.3830   0.3830
  Columns 12 through 22
  -0.0311  -0.0311  -1.1133 -0.3914  0.3305  -0.0792   0.2443
```

```
    -0.7011   -0.7393    0.3660    1.0000
  Columns 23 through 25
     0.8090    0.3090    0.8090
```

【例 35-10】

```
>>p0 = ppmak([0 1],[3 0 0]);
p1 = fn2fm(fn2fm(fnrfn(p0,[.4 .6]),'B-'),'pp')
p1 =
     form: 'pp'
     breaks: [0 1]
     coefs: [3 -4.4409e-016 2.2204e-016]
     pieces: 1
     order: 3
     dim: 1
```

注意：当把 B 样条函数转化为分段多项式样条函数时，由于第 1 个节点与最后一个节点（$t(1)$与 $t(n+k)$）的间断性，这两个节点将丢失，因为分段多项式样条函数在它的取值区间外受到限制。

例如，sp=spmak([0 1],1)在区间[0 1]内生成 B 样条函数，即：

```
sp =
    form: 'B-'
    knots: [0 1]
    coefs: 1
    number: 1
    order: 1
    dim: 1
```

然而，

```
pp=fn2fm(sp,'pp')
pp =
    form: 'pp'
    breaks: [0 1]
    coefs: 1
    pieces: 1
    order: 1
    dim: 1
```

5．fnder 函数

利用该函数求样条函数的微分（即求导数），其调用格式为：

- fprime = fnder(f)
- fprime = fnder(f, dorder)

fnder(f,dorder)表示对函数 f 求 dorder 阶导数，dorder 的默认值是 1。如果 dorder 是负值，表示对 f 函数求|dorder|重不定积分。

返回函数与输入函数的类型相同，即它们都是分段多项式样条函数或都是 B 样条函数。

如果函数 f 是多元函数，则 dorder 不能省略，dorder 要给出[d1,d2….dm]的形式。此时，对每个变量可求不同阶导数。

如果函数 f 是最后一个节点有多阶导数的 B 样条函数，则在允许误差范围内，函数 f 与

fnint(fnder(f))的返回函数相同。

如果函数 f 是分段多项式样条函数，则在允许误差范围内，函数 f 与 fnint(fnder(f))的返回函数相同，除非函数 f 有跳跃的不连续性。

如果函数 f 是 B 样条函数，t_1 是最左边的一个节点，则在允许误差范围内，fnint(fnder(f))的返回结果等同于 $f-f(t_1)$ 的结果。

sp=spmak([0 0 1],1)在区间[0 1]内是直线段，在 1 处函数值等于 0，而在 0 处函数值等于 1。spdi=fnint(fnder(sp))返回的样条函数在区间[0 1]内也是直线段，但是，在 0 处函数值等于 0，1 处函数值等于-1。

【例 35-11】

```
>>x=2*pi*[0 1 0.1 0.2 0.9];
y=cos(x);
cs=csapi(x,y);
fnder(cs)
ans =
form: 'pp'
   breaks: [0 0.6283 1.2566 5.6549 6.2832]
   coefs : [4x3 double]
   pieces: 4
   order: 3
dim:1
```

6. fndir 函数

用 fndir 函数求样条函数的方向导数，其调用格式为：

- df = fndir(f, direction)

方向导数的定义如下：假设矩阵 y 只有一列，则函数 df 为函数 f 在 y 方向的方向导数。即

$$D_y f(x) := \lim_{t->0}(f(x+ty)-f(x))/t$$

如果矩阵 direction 有 n 列，而函数 f 是 m 元 n 次的函数，则函数 df 是 m 元 n 次的函数。如果函数 f 是 m 元 n 次的函数，x 是定义域内的一个节点数组，则 reshape(fnval(fndir(f, eye(d)), x), m, d)为函数在 x 处的 Jacob 值。

【例 35-12】

```
>>xx = linspace(-.1,1.1,13);
yy = linspace(0,1,11);
[x,y] = ndgrid(xx,yy);
z = franke(x,y);
pp2dir = fndir(csapi({xx,yy},z),eye(2));
grads=reshape(fnval(pp2dir,[x(:) y(:)]'),[2,length(xx),length(yy)]);
quiver(x,y,squeeze(grads(1,:,:)),squeeze(grads(2,:,:)))
```

上述命令求出函数 Franke 的方向梯度图如图 35-2 所示。

7. fnint 函数

利用该函数求样条函数的积分，其调用格式为：

- intgrf = fnint(f)
- intgrf = fnint(f, value)

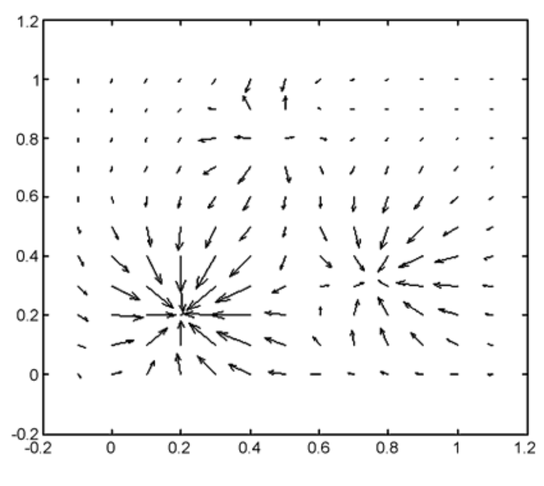


图 35-2　Franke 函数的方向梯度图

fnint(f, value) 求一元函数 f 的不定积分，参数 value 为求出的不定积分函数的常系数，默认值为 0。也就是说，如果求出的不定积分函数为 $F(x)=G(x)+C$，此时 C 就用参数 value 代替。返回的函数 intgrf 与输入函数 f 是相同形式的函数，即它们都是分段多项式样条函数或都是 B 样条函数。

如果函数 f 是分段多项式样条函数或是最右边的那个节点有足够高的重次的 B 样条函数，则在允许精度范围内 f 与 fnint(fnder(f)) 返回的结果相同，除非 f 函数有间断跳跃的不连续点。

如果 f 是 B 样条函数，t_1 是取值区间最左边的节点，则在允许精度范围内 fnint(fnder(f)) 的返回结果等于 $f-f(t_1)$ 的结果。

多元函数的不定积分可以通过设置函数 fnder(f, dorder) 中的参数 dorder 为负值而求得。

【例 35-13】　sp=spmak([0 0 1], 1)求得的样条曲线在区间[0　1]内是直线段，在右端点 1 处函数值等于 0，在左端点 0 处函数值等于 1，如图 35-3 所示。

spdi=fnint(fnder(sp)) 返回在区间[0　1]内的样条函数，但此函数在左端点 0 处等于 0，在右端点 1 处等于-1，如图 35-4 所示。

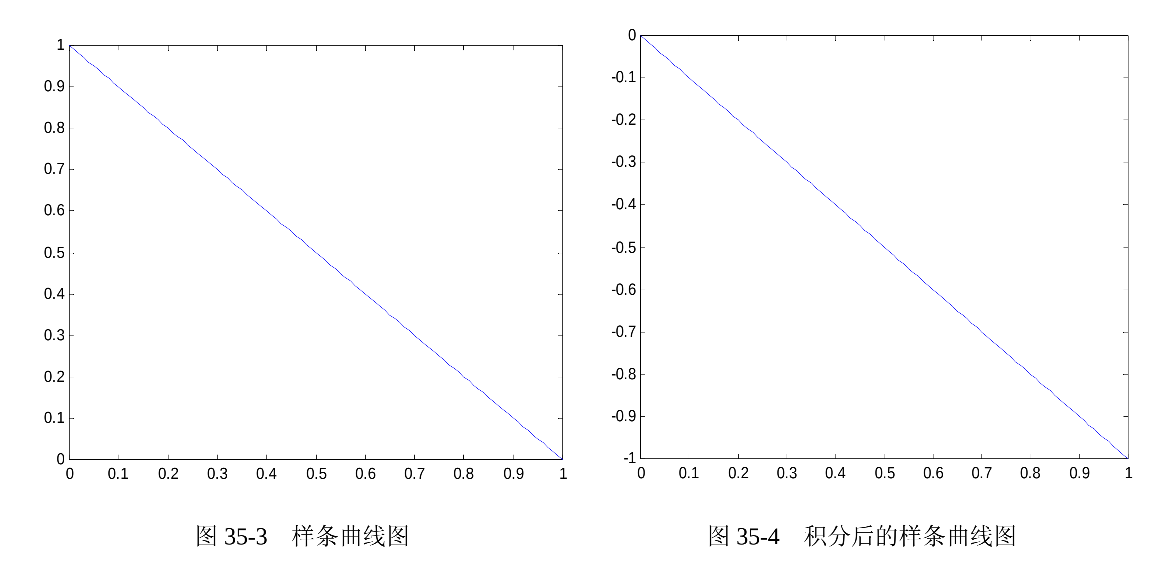


图 35-3　样条曲线图

图 35-4　积分后的样条曲线图

8．fnjmp 函数

利用该函数在间断点处求函数值，其调用格式为：

- jumps = fnjmp(f,x)。它返回函数 f 在间断点 x 处 $f(x+)-f(x-)$的函数值，此处要求函数 f 必须为一元函数。如果 $\boldsymbol{x}$ 是一个矩阵形式，则输出参数 jumps 也是一个矩阵，矩阵大小与 $\boldsymbol{x}$ 相同。

【例 35-14】

```
>>fnjmp(ppmak(1:4,1:3),1:4)
ans =
     0    1    1    0
```

因为在区间[1 2]内分段多项式样条函数值是 1，在区间[2 3]内函数值是 2，在区间[3 4]内函数值是 3，因此，由 $f(x+)-f(x-)$计算得到在 1 和 4 处为 0，在 2 和 3 处为 1。

又如：

```
>>x=cos([4:-1:0]*pi/4);
fnjmp(fnder(spmak(x, 1), 3), x)
ans =
     32.0000  -24.0000   24.0000  -24.0000   32.0000
```

若把函数转化成分段多项式样条函数：

```
>>x=cos([4:-1:0]*pi/4); fnjmp(fnder(fn2fm(spmak(x, 1), 'pp'), 3), x)
ans =
     0  -24.0000   24.0000  -24.0000        0
```

这个结果与 B 样条函数结果不一样，这是因为分段多项式样条函数在它的取值区间的两端点处是连续的。

注意，看下面的例子

```
>>x=cos([4:-1:0]*pi/4);
fnjmp(fnder(spmak(x,1),3),-x)
ans =
     32.0000      0      0        0      32.0000
```

因为 $-x$ 与 x 在四舍五入成整数时不一样，因此三次 B 样条函数 spmak(x, 1）在 $-x(2)$, $-x(3)$, $-x(4)$ 处没有间断。

9．fnplt 函数

利用该函数画样条曲线图，其调用格式为：

- fnplt(f)
- fnplt(f, arg1, arg2, arg3, arg4)
- points = fnplt(f)

在给定的区间[a b]内，使用给定的画图符号、线宽画样条曲线图，其中参数 arg1 表示给定的画图符号，默认设置是‘−’，arg2 表示给定的区间，默认区间是[0 1]，arg3 表示画图所用的线宽，默认值是 1，上述 3 个参数的顺序可任意给定。若函数发生跳跃，在跳跃点处的函数值为 NaNs 时，就要用给定参数 arg4 才能画出图形。

画图主要取决于函数 f 是一元函数还是多元函数以及函数的阶次，即看它是一次的、二次的还是 d 次的函数（d>2)。

如果函数 f 是一元函数，则按下述规则画图。

如果函数 *f* 是一次函数，则返回的是函数 *f* 的曲线图。如果函数 *f* 是二次函数，则返回的是函数 *f* 的曲面图，如果函数 *f* 是 *d* 次函数(d>2)，则返回的是函数 *f* 的多维曲面图。

如果函数 *f* 是二元函数，则按下述规则画图。

如果函数 *f* 是一次函数，则返回的是函数 *f* 的曲线图。如果函数 *f* 是二次函数，则返回的是函数 *f* 投影在主平面上的曲面图。如果函数 *f* 是 *d* 次函数(d>2)，则返回的是函数 *f* 的表面图。

如果函数 f 是多元函数(大于二元)，则通过选取每个自变量取值区间中点的值来画图。

调用格式 points = fnplt(f)，不在屏幕上画出图形，而是将画图用的二维点或三维的数据点存储在数组 points 中。

【例 35-15】

```
>>x=2*pi*[0 1 0.1 0.2 0.9];
y=cos(x);
cs=csapi(x,y);
fnplt(cs)
```

返回结果如图 35-5 所示。

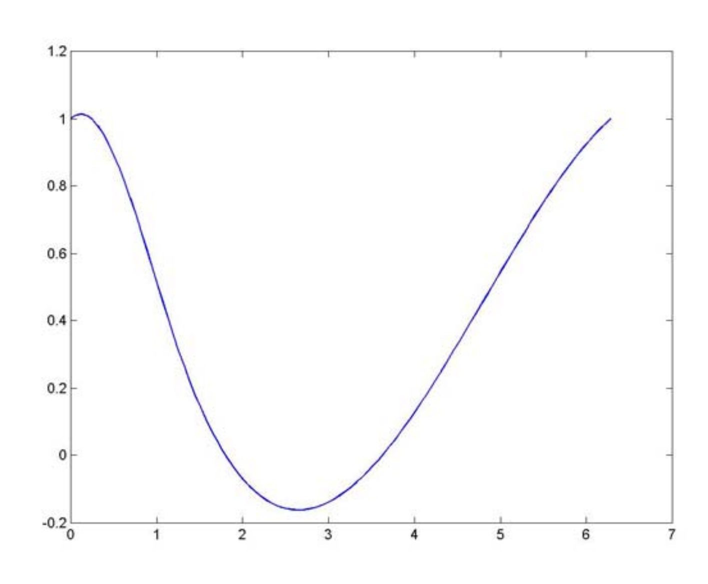


图 35-5　拟合的余弦函数样条曲线图

注意：B 样条函数的区间是包含所有节点的取值区间。这意味着，B 样条函数 *f* 的曲线在左右端点处终止，除非这两个端点都有 k 阶导数，此处的 *k* 为函数 f 的最高次幂（阶次）。

10．fnrfn 函数

利用该函数在样条曲线中插入节点，其调用格式为：

- g = fnrfn(f,addpts)。此函数将样条函数 *f* 的曲线形状通过插入的节点（断点）向量而被更精确地确定下来。当两个或多个不同类型的函数求和，或者通过增加样条函数的自由度以使局部变化拟合得更精确时，通常会使用该函数。

如果函数 *f* 是 B 样条函数或 BB 样条函数，则新加入的节点 addpts 插入已存在的节点向量中，而且必须保证没有一个节点的重次大于样条函数 *f* 的阶次。插入节点后的 B 样条函数

保存在输出参数 g 中。

如果函数 f 是分段多项式样条函数，则新加入的节点 addpts 插入已存在的节点向量中，而且必须保证所有的节点仍然是严格单调递增。插入节点后的分段多项式样条函数保存在输出参数 g 中。

如果函数 f 是多元函数，则新加入的节点 addpts 必须是数组[addts1,…,addtsm]。如果插入的数组 addpts 中，第 i 个元素为空，则相应的第 i 个变量的节点（断点）值不变。

【例 35-16】

```
>>p0 = ppmak([0 1],[3 0 0])
p0 =
     form: 'pp'
     breaks: [0 1]
     coefs: [3 0 0]
     pieces: 1
     order: 3
dim: 1
```

而

```
>>p0 = ppmak([0 1],[3 0 0]);
fnrfn(p0,[.4 .6])
ans =
     form: 'pp'
     breaks: [0 0.4000 0.6000 1]
     coefs: [3x3 double]
     pieces: 3
     order: 3
     dim: 1
```

11．fntlr 函数

该函数生成 Taylor 系数或 Taylor 多项式，其调用格式为：

- taylor = fntlr(f, dorder, x)
- p = fntlr(f, dorder, x, interv)

taylor = fntlr(f, dorder, x) 返回函数 f 在 x 处的非正规化 Taylor 系数，Taylor 系数的最高阶导数为 dorder 阶。

如果 f 是一元函数，x 是标量值，则返回的 Taylor 系数为

$$(f(x); df(x); \cdots; d^{\text{dorder}-1} f(x))$$

如果 $\boldsymbol{x}$ 是矩阵，则相应返回的 Taylor 系数也是一个矩阵形式。

如果 f 是多元函数，则要求 Taylor 偏导数的阶 dorder 也是数组。如果函数 f 是 m 元函数（m>1），此时要求数组 dorder 的长度也是 m。假设 dorder 数组是(i1, i2, …, im)，则对第 j 个输入变量 $\boldsymbol{x}(:, j)$（$\boldsymbol{x}$ 是矩阵）的 Taylor 系数输出为

$$d_1^{i_1-1} d_2^{i_2-1} \cdots d_m^{i_m-1} f(x)$$

输出第 j 个 Taylor 系数 taylor(:,j)的长度是 m*i1*i2*…*im。

p = fntlr(f,dorder,x, interv) 返回函数 f 在 x 处的分段多段项式 Taylor 系数。分段多项式的区间由参数 interv 决定。 如果函数 f 是 m 元函数，则要求矩阵 $\boldsymbol{x}$ 大小为 size[m,1]，参数 interv

是大小为 size[m,2]的矩阵，或者是包括 *m* 个大小为 size[1,2]向量的数组。

如果函数 *f* 是一元函数，*x* 是标量或是只有一行的矩阵，则 fntlr(f,3,x) 与下面的输出结果相同：

```
df = fnder(f); [fnval(f,x); fnval(df,x); fnval(fnder(df),x)];
```

【例 35-17】

```
>>ci = rsmak('circle');
in = fnbrk(ci,'interv');
t = linspace(in(1),in(2),21);
t(end)=[];
v = fntlr(ci,3,t);
%在圆上对点 v(1:2,j) 用'o'作标记
fnplt(ci), hold on, plot(v(1,:),v(2,:),'o');
quiver(v(1,:),v(2,:),v(3,:),v(4,:)); %用箭头标出点 v(1:2,j)的切向量
%用用箭头标出点 v(1:2,j)的二阶导数向量
quiver(v(1,:),v(2,:),v(5,:),v(6,:)), axis equal, hold off
```

结果如图 35-6 所示。

下面是一个两元 Runge 函数的例子。

```
>>w = csapi({-1:1, -1:1},[3 2 3;2 1 2;3 2 3]);   %内插生成函数 1/(1+x^2+y^2)
wcoefs = fnbrk(w,'coef');
scoefs = zeros(size(wcoefs)); scoefs(end)=1;
runge2 = rpmak(fnbrk(w,'breaks'),[scoefs; wcoefs]);   %生成有理样条曲线
fnplt(fnbrk(runge2, {[-2 2], [-2 2]}));
shading interp, hold on
fnplt(fntlr(runge2,[3 3], [0;0],[-5 .5; -.5 5])), hold off
```

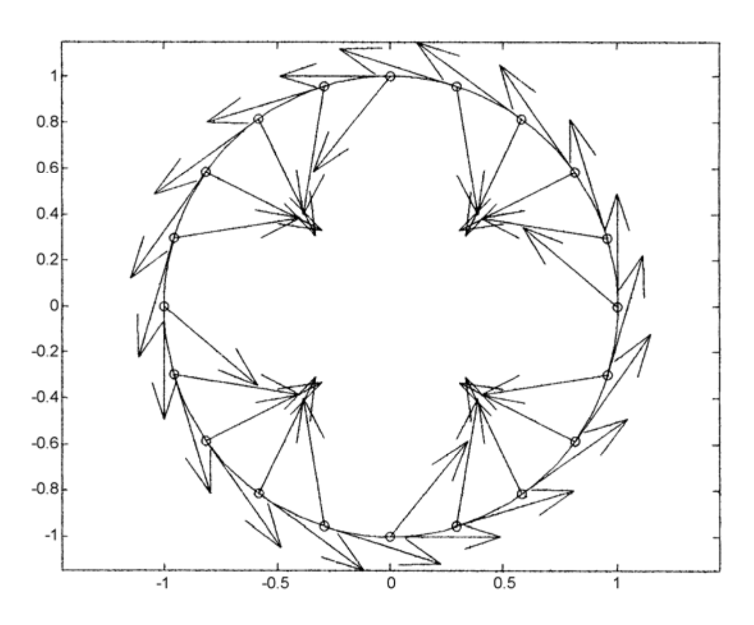


图 35-6　有理样条曲线一阶与二阶导数图

结果如图 35-7 所示。

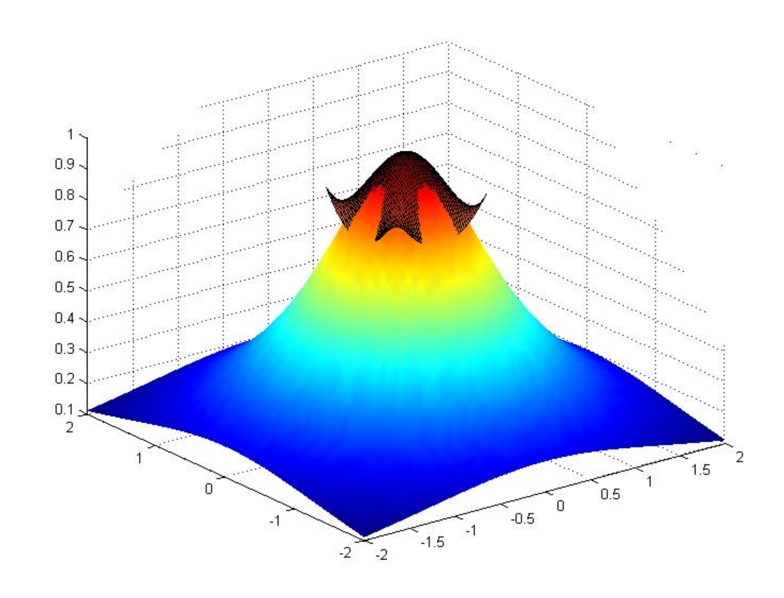


图 35-7　函数 1/(1+x^2+y^2)与它的在起始点处的[3,3]阶 Taylor 多项式曲线图

为了生成函数 1/(1+x^2+y^2) 在起始点处的 3 阶 Taylor 多项式，执行：

```
>>taylor = fntlr(runge2,[3 3],[0;0],[0 1;0 1]);
tcoef = fnbrk(taylor,'coe');
tcoef([1 2 4]) = 0;
taylor2 = fnbrk(ppmak(fnbrk(taylor,'br'),tcoef),{[-1 1],[-1 1]});
fnplt(fnbrk(runge2,{[-2 2],[-2 2]}));
shading interp, hold on
fnplt(taylor2), view(-28,-26), axis off, hold off
```

结果如图 35-8 所示。

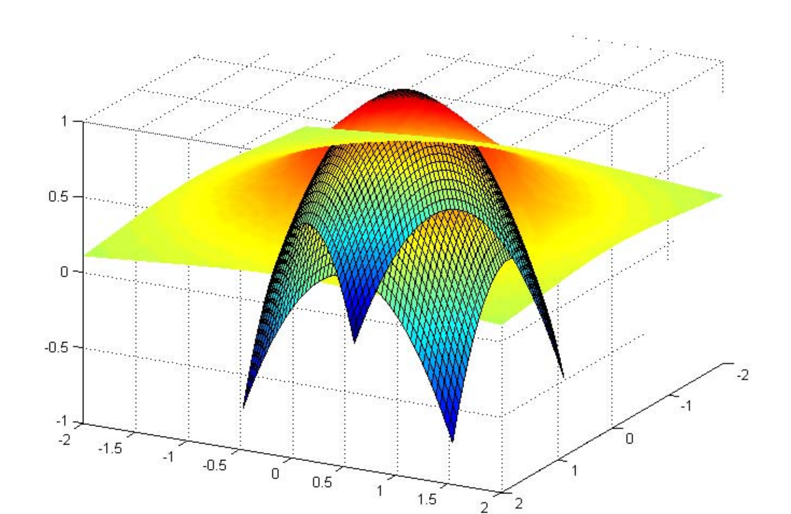


图 35-8　函数 1/(1+x^2+y^2)与它的在起始点处的 3 阶 Taylor 多项式曲线图

第 36 章　样条曲线的端点与节点处理类函数

MATLAB 样条工具箱中提供的端点与扭结点处理类函数如表 36-1 所示。

表 36-1　样条曲线的端点与扭结点（节点）处理类函数

函数名称	简 单 介 绍
augknt	在已知节点数组中添加一个或多个节点
aveknt	求出节点数组元素的平均值
brk2knt	增加断点数组中元素的重次
knt2brk	从节点数组中求得节点及其重次
knt2mlt	从节点数组中求得节点及其重次
sorted	求出节点数组 points 的元素在节点数组 meshpoints 中属于第几个分量
aptknt	求出用于生成样条曲线的节点数组
newknt	对分段多项式样条函数进行重分布
optknt	求出用于内插的最优节点数组
chbpnt	求出用于生成样条曲线的合适节点数组

下面按表 36-1 所列，对每个函数逐一进行介绍。

1．augknt 函数

该函数在已知节点数组中添加一个或多个节点，其调用格式为：

- [augknot, addl] = augknt(knots, k)
- [augknot, addl] = augknt(knots, k, mults）

此函数返回单调递增的已添加了一个或多个节点的节点数组，该节点数组中最大与最小元素的重次为 *k* 重。此处，重次表示某个元素重复出现的次数，例如数组[1 2 2 2 3]中元素 2 的重次为 3。一般地，重次等于最高阶导数的阶次，因此重次越高，曲线在该节点的连续性越高。此外，该函数也可以返回添加到节点数组取值区间左边的节点个数（此值有可能是负的）。

如果给定的第 3 个输入参数 mults 是标量，则节点数组内的每个元素（两个端点元素除外）是 mults 重的。如果参数 mults 是向量，并且向量 mults 的元素个数与节点数组（两端点元素除外）的元素个数相等，则节点数组中的第 *j* 个元素为 mults(j) 重。如果向量 mults 的元素个数与节点数组（两端点元素除外）的元素个数不相等，则每个元素都为 mults(1) 重。如果节点数组严格单调递增，则由 *k* 阶节点数组 augknot 生成的样条曲线在 knots(j+1)处的重次为 k-mults(j)，j=1:（length(knots)-2）。

【例 36-1】　如果要在区间[*a*　*b*]内生成一条三次样条曲线，有两阶连续偏导，并有一个内部断点 xi，则可以用 augknt([a, b, xi], 4)来生成需要的节点向量。例如，

```
>>a=1;
b=2;
xi=1.5;
```

```
augknt([a,b,xi],4)
ans =
    1.0000    1.0000    1.0000    1.0000    1.5000
    2.0000    2.0000    2.0000    2.0000
```

如果使用 Hermite 插值法生成三次样条函数，即生成只有一阶连续偏导的三次样条函数，则可以用 augknt([a, xi, b], 4, 2) 来生成需要的节点向量。例如：

```
>>a=1;
b=2;
xi=1.5;
augknt([a,xi,b],4,2)
ans =
    1.0000    1.0000    1.0000    1.0000    1.5000
    1.5000    2.0000    2.0000    2.0000    2.0000
```

又如：

```
>>a=1;
b=2;
xi=1.5;
x1=1.8;
augknt([a,xi,x1,b],4,[2 3])
ans =
Columns 1 through 7
    1.0000    1.0000    1.0000    1.0000    1.5000    1.5000    1.8000
Columns 8 through 13
    1.8000    1.8000    2.0000    2.0000    2.0000    2.0000
```

2．aveknt 函数

用该函数求出节点数组元素的平均值，其调用格式为：

- tstar = aveknt(t,k)。此函数返回连续的$(k-1)$个节点的平均值，即

$$t_i^* = (t_{i+1} + \ldots + t_{i+k-1})/(k-1),\ i = 1, \ldots n$$

当在有 k 个元素的节点数组中内插值时，通过此函数可以找出最佳插值点。

【例 36-2】

```
>>aveknt([1 2 3 3 3], 3)
ans =
    2.5000    3.0000
```

对于 k 阶严格单调递增的断点向量，

```
t = augknt(breaks,k);
x = aveknt(t,kk);
sp = spapi(t,x,sin(x))
```

上面返回在区间[breaks(1) breaks(end)]内插值生成的正弦样条函数。例如，

```
>>t = augknt([0 2 4],4);
x = aveknt(t,4);
sp=spapi(t,x,sin(x));
fnplt(sp)
```

结果如图 36-1 所示。

3．brk2knt 函数

利用该函数增加断点数组 breaks 中元素的重次，其调用格式为：

- [knots, index] = brk2knt(breaks, mults)

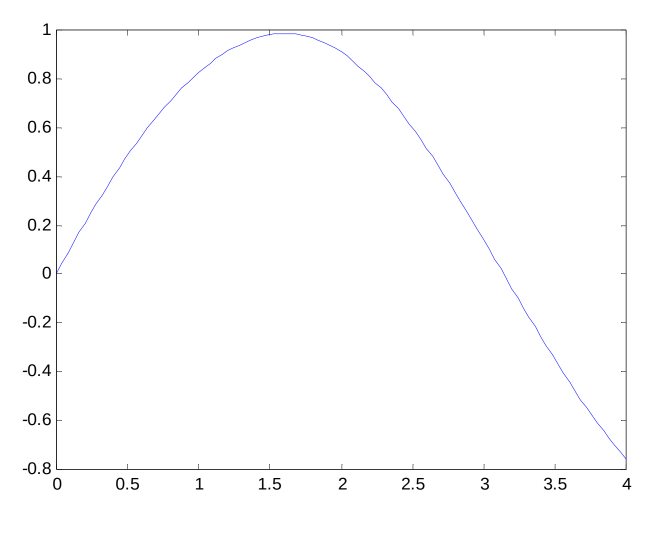


图 36-1　B 样条曲线图

输入参数 breaks 为断点数组，mults 为与 breaks 相对应的重次数组。特别地，若 mults(i)<=0，则第 breaks(i) 断点元素不增加重次。若断点数组 breaks 严格单调递增，则 knots 为断点数组 breaks 中的每个元素增加 mults 重后生成的节点数组。

若数组 mults 的元素个数与断点数组 breaks 的元素个数不相等，则所有的 mults(i)等于 mults(1)。

若断点数组 breaks 严格单调递增，而且所有的 mults(i)为正，则对每个 i，index(i) 是断点数组元素 breaks(i)在生成的节点向量 knots 中的下标值。

【例 36-3】

```
>>xi=[1 2 3 4 5];
m=[2 3 1 1 2];
tt=brk2knt(xi,m)
tt =
     1     1     2     2     2     3     4     5     5
```

4．knt2brk, knt2mlt 函数

该函数从节点数组中求得节点及其重次，其调用格式为：

- [breaks, mults] = knt2brk(knots)
- [m, sortedt] = knt2mlt(t)

函数 knt2brk 与 knt2mlt 用于从节点数组中求得不相重复的节点元素和它们的重次。在两个不同的函数中，重次略有差别。

函数 knt2brk(knots)返回节点数组 knots 的元素及其重次。数组 breaks 与 mults 的长度是相等的，可选参数 mults 存放节点数组中每个元素的重次。函数 knt2brk 是函数 brk2knt 的反操作，对于任意的节点数组，[xi, mlts]=knt2brk(knots); knots1=brk2knt(xi, mlts)求出的节点数组 knots1 等于 knots。

函数 m=knt2mlt(t)返回与数组 t 长度相等的数组 m，m(i)等于第 i 个节点的重次。注意，数组 t 已经过函数 sort(t)排序。求出的数组 m 可作为函数 spapi 或 spcol 函数的输入参数。特别地，如果数组 t 单调递增，z 是与数组 t 长度相等的一个数组，则 sp=spapi(knots, t, z)可以生成样条函数 sp。此处，对所有的 i，满足 $D^{m(i)}s(t(i)) = z(i)$。第 2 个可选输出参数 sortedt 返回函数 sort(t)的结果。

一般用户很少用到这两个函数。

【例 36-4】

```
>> [xi,mlts]=knt2brk([1 2 3 3 1 3])
xi =
    1    2    3
mlts =
    2    1    3
```

又如

```
>> [m, t]=knt2mlt([1 2 3 3 1 3])
m =
    0    1    0    0    1    2
t =
    1    1    2    3    3    3
```

5. sorted 函数

利用该函数求出节点数组 points 中元素在节点数组 meshpoints 中属于第几个分量，其调用格式为：

- index = sorted(meshpoints, points)

在样条工具箱中，有许多的 M 文件要确定数组中某个元素在数组中的位置。例如，确定元素 x 在数组$[t_j \ldots t_{j+1}]$中的位置，注意数组 t 是一单调递增的节点数组。此时，可以通过此函数对元素 x 进行定位，求出元素 x 在数组 t 中的位置编号。

index = sorted(meshpoints, points)求得的位置编号数组 index 是一个整数数组。index(j)可以按如下来确定：首先，spoints=sort(points)，然后求出数组 meshpoints 中小于等于 spoints(j)的元素个数，求出的值即为 index(j)。因此，如果数组 meshpoints 与 points 都单调递增，则

$$\text{meshpoints(index(j))} \leqslant \text{points(j)} < \text{meshpoints(index(j)+1)}$$

此处，index(j) < 1 或者 length(meshpoints) < index(j) + 1

【例 36-5】

```
>>sorted([1 1 1 2 2 3 3 3],[0:4])
ans =
    0    3    5    8    8
```

同样

```
>>sorted([3 2 1 1 3 2 3 1],[2 3 0 4 1])
ans =
    0    3    5    8    8
```

6. aptknt 函数

利用该函数求出用于生成样条曲线的节点数组，其调用格式为：

- knots = aptknt(tau, k)

● [knots, k] = aptknt(tau, k)

假设数组 tau 的元素个数大于等于 k，并且数组 tau 单调递增，对所有的 i，满足 tau(i)<tau(i+k-1)，则 aptknt(tau, k)返回的 knots 可以作为内插生成样条曲线的节点数组，即返回的节点数组 knots 满足 Schoenberg-Whitney 条件：

knots(i) < tau(i) < knots(i+k), i=1:length(tau),

如果数组 tau 的元素个数少于 k，则忽略输入参数 k，并且返回 k=length(tau)。如果数组 tau 不是单调递增的，或者对某个 i，tau(i)=tau(i+k-1)，则有出错消息返回。

实际上，只要数组 tau 的元素个数大于等于 k，k>1，则 aptknt(tau, k)返回的节点数组 knots 与 augknt([tau(1), aveknt(tau, k), tau(end)], k)返回的节点数组相等。例如：

```
>>tau=[3 4 5 6 7 10];
k=5;
augknt([tau(1),aveknt(tau,k),tau(end)],k)
ans =
    3.0000    3.0000    3.0000    3.0000    3.0000
    5.5000   30.0000   30.0000   30.0000   30.0000   30.0000
```

下面，

```
>>aptknt([3 4 5 6 7 10],5)
ans =
    3.0000    3.0000    3.0000    3.0000    3.0000
    5.5000   30.0000   30.0000   30.0000   30.0000   30.0000
```

【例 36-6】

```
>>tau = linspace(1,10,4);
y=cos(tau);
sp=spapi(aptknt(tau,4),tau,y);
fnplt(sp)
```

结果如图 36-2 所示。

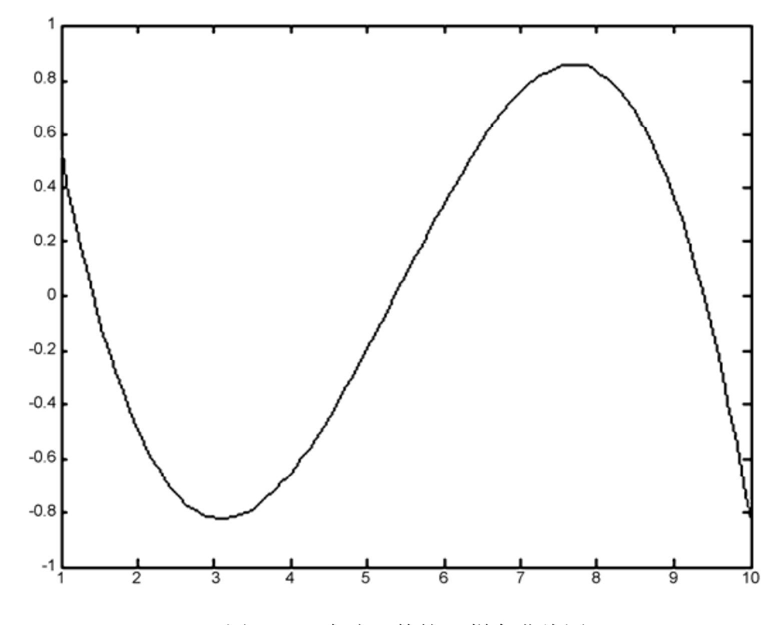


图 36-2 余弦函数的 B 样条曲线图

除了返回的是分段多项式样条曲线外，下面返回的样条曲线与上面完全一样。

```
>>tau = linspace(1,10,4);
y=cos(tau);
sp=spline(tau,y);
fnplt(sp)
```

结果如图 36-3 所示。

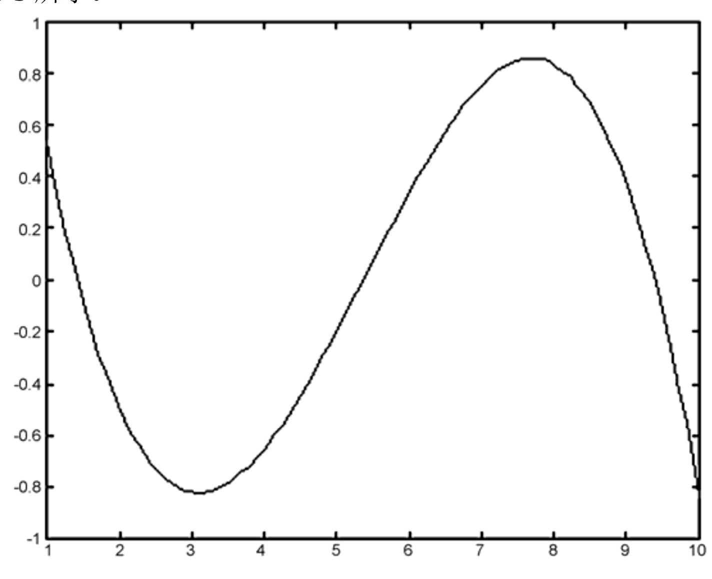


图 36-3　余弦函数的分段多项式样条曲线图

实际上，仅仅从样条曲线上根本看不出它们的区别。

7．newknt 函数

利用该函数对分段多项式进行重分布，其调用格式为：

- newknots = newknt(f)
- newbreaks = newknt(pp, newl)
- [newbreaks, distfn] = newknt(pp, newl)

此函数把分段多项式样条函数分成 newl 段。同时，pp 的取值区间也被分成 newl 段。函数返回各段分段多项式的取值区间，如果需要返回生成的分段多项式函数，可以输入参数 distfn。通过此函数生成了一个与 pp 的高阶导数相关的分段多项式线性单值函数。如果输入参数 newl 没有给出，则原先函数 *f* 的多项式段数作为默认的 newl。

【例 36-7】

```
>>x=linspace(0,10,101);
y=exp(x);
sp0=spap2(augknt([0:2:10],4),4,x,y);
[newbreaks,distfn]=newknt(sp0,5)
newbreaks =
Columns 1 through 11
   0         0        0         0          4.4150    6.3435
   7.6624    8.8533   30.0000   30.0000   30.0000
Column 12
   30.0000
```

```
distfn =
     form: 'pp'
     breaks: [0 2 4 6 8 10]
     coefs: [5x2 double]
     pieces: 5
     order: 2
dim: 1
```

8. optknt 函数

利用该函数求出用于内插的最优节点数组，其调用格式为：

- knots = optknt(tau,k)
- knots = optknt(tau, k, maxiter)

t=optknt(tau, k) 从数组 tau 中选出用于内插生成 k 次样条曲线的最优节点数组。输入参数 maxiter 为计算时的迭代次数，默认值为 10。此处，“最优”的意思是：

在元素 tau(1),…,tau(n) 中，求出与向量 g 匹配的最优内插向量 Rg，这里定义最小常数 constR，使得对所有的 g，满足||g - Rg||≤constR||D^kg||。

$\|f\| := \sup_{tau(1)<x<tau(n)}|f(x)|$，从中找出最小的 constR，Micchelli/Rivlin/Winograd 已经表明了这种关系, t 由下面的条件决定。

（1）t(1) = ... = t(k) = tau(1);

（2）t(n+1) = ... = t(n+k) = tau(n);

（3）t(k+1), ..., t(n) 满足下面的函数 *h*,

$$\int_{tau(1)}^{tau(n)} f(x)h(x)\mathrm{d}x = 0, \qquad \text{所有的 } f \in S_{k,t}$$

【例 36-8】

```
>>x=2*pi*sort([0 1 rand([1 10])]);
optknt(x,5,20)
ans =
Columns 1 through 11
   0        0        0        0        0        1.3658
   1.9761   2.7022   3.5391   4.3489   4.9944
Columns 12 through 17
   5.4521   6.2832   6.2832   6.2832   6.2832   6.2832
```

9. chbpnt 函数

利用该函数求出用于生成样条曲线的合适节点数组，其调用格式为：

- tau = chbpnt(t, k)
- [tau, sp] = chbpnt(t, k, tol)

函数 chbpnt 从节点数组 t 中求出生成 k 阶 Chebyshev 样条函数的合适节点数组。返回的 tau 是内插生成 k 阶 Chebyshev 样条函数的最优节点数组。Chebyshev 样条函数由 Remes 算法迭代生成。

如果输出参数中有 sp，则参数 sp 存放返回的 chebyshev 样条函数的信息。输入参数 tol 是允许的迭代精度，默认是 0.001。在生成 Chebyshev 样条函数时，如果前后两次迭代的差值小于 tol 值，就结束迭代，生成满足精度要求的 Chebyshev 样条函数。

【例 36-9】 chbpnt([-ones(1, k),ones(1, k)], k)在区间[-1 1]生成自由度为 k-1 的 Chebyshev 多项式。例如：

```
>>k=5;
[tau, sp]=chbpnt([-ones(1, k),ones(1, k)], k)
tau =
   -1.0000   -0.7071   -0.0000    0.7071    1.0000
sp =
     form: 'B-'
     knots: [-1 -1 -1 -1 -1 1 1 1 1 1]
     coefs: [-1.0000 7.0000 -31.6667 7.0000 -1]
     number: 5
     order: 5
     dim: 1
```

【例 36-10】

```
>>k = 4;
n = 10;
t = augknt(((0:n)/n).^8,k);
x = chbpnt(t,k);
sp = spapi(t,x,sqrt(x));
```

结果如下所示：

```
sp =
    form: 'B-'
    knots: [1x17 double]
    coefs: [0 8.8692e-005 0.0012 0.0056 0.0175 0.0429 0.0902 0.1703 0.2963
           0.4837 0.7520 0.9040 1]
   number: 13
   order: 4
   dim: 1
```

其图形如 36-4 所示。

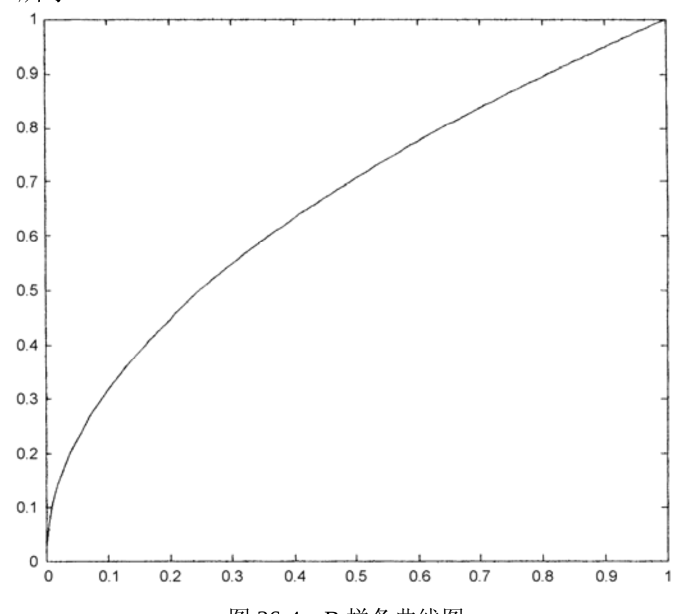


图 36-4　B 样条曲线图

第 37 章　解线性方程组类函数

MATLAB 样条工具箱中提供的解线性方程组类函数如表 37-1 所示。

表 37-1　解线性方程组类函数

函 数 名 称	简 单 介 绍
slvblk	解对角占优的线性方程组
bkbrk	描述分块对角矩阵的详细情况

1．slvblk 函数

该函数解对角占优的线性方程组，其调用格式为：

- x = slvblk(blokmat, b)
- x = slvblk(blokmat, b, w)

slvblk(blokmat,b)返回线性方程组 $Ax=b$ 的解，输入参数 blokmat 存放矩阵 $\boldsymbol{A}$，它是对角占优的矩阵，矩阵中元素的典型值是 B 样条函数在一些坐标点处的方差值（包括第零阶方差，即点的坐标值）。

如果线性方程组是超静定的，则该函数利用最小二乘法求解。此时，应给出可选参数 w，以确保 $l_2\sum\sum jw(j)*((A*x-b)(j))^2$ 的值最小。式中，b 是一包含多行的单列矩阵，其行数与矩阵 blokmat 的行数相等。

【例 37-1】

```
>>knots=[1 2 3 4 5 6];
k=3;
x=linspace(1,6,10);
y=sin(x);
yy=y';
slvblk(spcol(knots,k,x,1),yy)
ans =
    1.1639
   -0.4666
   -1.4005
```

sp=spmak(knots, slvblk(spcol(knots, k, x, 1), y.'))返回与数组（x, y）相匹配的 k 阶 B 样条函数 s，即满足 s(x)=y。例如，

```
>>knots=[1 1 3 4 5 7];
k=3;
x=linspace(1,6,8);
y=sin(x);
sp=spmak(knots,slvblk(spcol(knots,k,x,1),y.'))
sp =
     form: 'B-'
     knots: [1 1 3 4 5 7]
     coefs: [3x1 double]
```

```
number: 1
order: 5
dim: 3
```

2．bkbrk 函数

该函数返回分块对角矩阵的详细情况，其调用格式为：

- [nb, rows, ncols, last, blocks] = bkbrk(blokmat)
- bkbrk(blokmat)

函数bkbrk用于函数slvblk中，它返回分块对角矩阵blokmat的详细情况，包括行数rows、分块矩阵的大小size[sum(rows), ncols]等。

如果没有输出参数，则有关分块矩阵的详细情况显示在屏幕上。这对于弄清分块矩阵哪里出问题非常有用。

【例 37-2】

```
>>t=linspace(0,2*pi,10);
x = linspace(t(1),t(end),10);
c = spcol(t,3,x,'sl');
[NB,ROWS,NCOLS,LAST,BLOCKS]=bkbrk(c)
NB =
    7
ROWS =
    3    1    1    1    1    1    2
NCOLS =
    3
LAST =
    1    1    1    1    1    1    1
BLOCKS =
         0         0         0
    0.5000         0         0
    0.5000    0.5000         0
    0.5000    0.5000         0
    0.5000    0.5000         0
    0.5000    0.5000         0
    0.5000    0.5000         0
    0.5000    0.5000         0
    0.5000    0.5000         0
    0         0             1.0000
```

第 38 章　样条 GUI 函数

为了使用户能够方便地使用上面介绍的函数，MATLAB6.0的样条工具箱中新增了两个样条GUI函数bspligui和splinetool。下面介绍这两个函数。

1．bspligui 函数

该函数在节点处生成B样条曲线，其调用格式为：

- bspligui。它演示在 GUI（图形用户界面）下，如何根据已知的节点生成一条 B 样条曲线。用户可以用鼠标在生成的 B 样条曲线上增加、移动或删除节点。当用户进行这些操作时，B 样条曲线与它的一阶、二阶、三阶导数曲线也随之发生变化。

用户在MATLAB工作空间中输入bspligui，回车后，就显示图38-1所示的GUI界面。

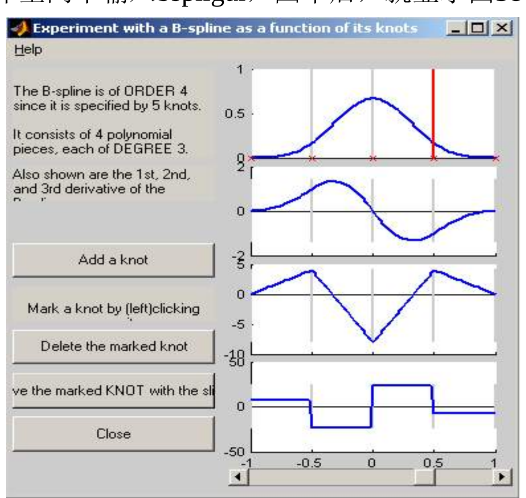


图38-1　bspligui函数的GUI界面

图38-2为用鼠标在图38-1上随意点击，增加3个节点后的情况。

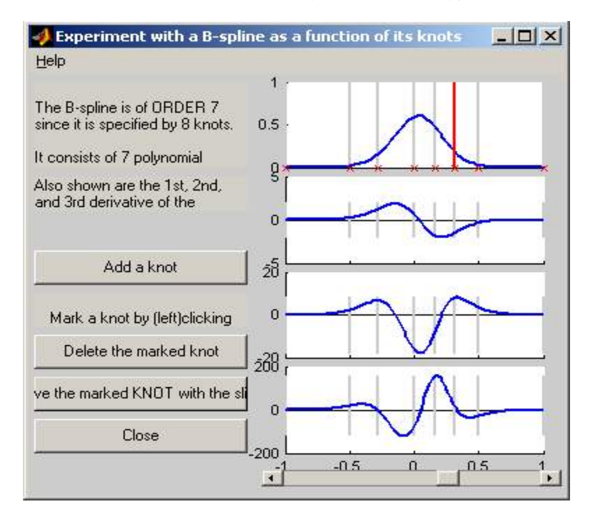


图 38-2　增加三个节点后的 bspligui 函数的 GUI 界面

通过分析上面两图在节点数组$t_0 \leqslant \cdots \leqslant t_k$ 内生成的B样条曲线，可以得到以下结论。

（1）B 样条函数值在开区间(t_0,t_k)内为正。若节点 t_0、t_k 不是 k 重节点，则在节点 t_0、t_k 处，函数值为零。在区间$[t_0,t_k]$外，B 样条函数值也为零，但在区间外的曲线部分，界面上没有显示出来。

（2）即使在最大值处，B 样条函数值也比 1 小。在区间内，只有节点至少为（k-1）重时，函数值才有可能为 1。另外，最大值也不能任意小，当内部没有节点时，函数值就会非常小。

（3）B 样条曲线实质上是一条 k 阶分段多项式样条曲线，即对于 k=1:4，各段的自由度 degree<k。实际上，通过观察它的一阶、二阶、三阶导数曲线，可以看出它的所有分段多项式样条曲线的自由度 degree 都是（k-1）。这意味着，每当增加或减少一个节点时，自由度随之升高或降低一度。

（4）允许 B 样条曲线的几个节点重合。因此，分段多项式的段数最多为 k 段（每个节点都不相同），最少为 1 段（所有的节点都相等），即样条曲线的阶次在 1 与 k 之间变化。

（5）B 样条曲线的光滑程度依靠各节点的重次。如果某个节点在节点数组中出现 m 次，则 B 样条曲线在此节点处有（k-m）阶偏导数，所有比（k-m）阶低的偏导数，在此节点处是连续的。因此，随着一个节点重次的变化，可以控制 B 样条曲线的光滑程度。

（6）当一个节点与另一个节点离得非常近时，在两节点间的最高阶偏导数有一个巨大的跳跃，而且最高阶偏导数成为无界。但是，低阶偏导数没有这种情况发生。

（7）B 样条曲线的形状变化规律如下：

如果在区间(t_0, t_k)内，一阶偏导数不恒为零，则一阶偏导数曲线有一次正负号变化。因此，B 样条曲线本质上是单模式的，即有一个确定的最大值。如果在区间(t_0, t_k)内，二阶偏导数曲线不恒为零，则二阶偏导数曲线有两次正负号变化。如果在区间(t_0, t_k)内，三阶偏导数曲线不恒为零，则三阶偏导数曲线有三次正负号变化。这表明，如果第 n 阶偏导数曲线在区间内不是恒为零，则此偏导数曲线有 n 次正负号变化。

2．splinetool 函数

该函数用一系列方法生成各种样条曲线，其调用格式为：

- splinetool
- splinetool(x, y)

splinetool函数的功能非常强大，它几乎把前面各种生成样条曲线的方法都包括在内了。splinetool函数提供一个图形用户界面（GUI），它的初始菜单提供了各种数据，用户可以从中选择一种，生成需要的样条曲线。

splinetool(x, y) 根据数组x，y在图形用户界面下生成样条曲线，但要求输入的数组x与y的长度相等。

用户在 MATLAB 工作区空间内输入 splinetool，回车后，就显示图 38-3 所示的界面。

从中选择一种数据，例如，选择 Noisy values of a smooth function，就显示图 38-4 所示的 GUI 界面。

从图形用户界面的 Approximation 项中，可以选择各种生成样条曲线的方法。可供选择的方法及其可调情况如表 38-1 所示。

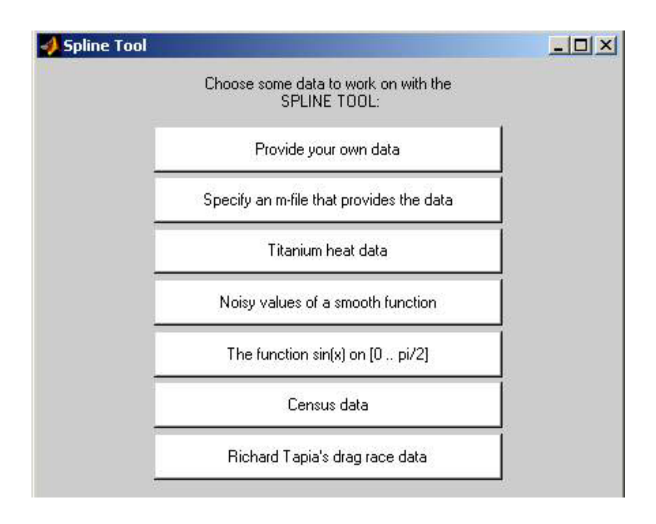


图 38-3　splinetool 函数的初始界面

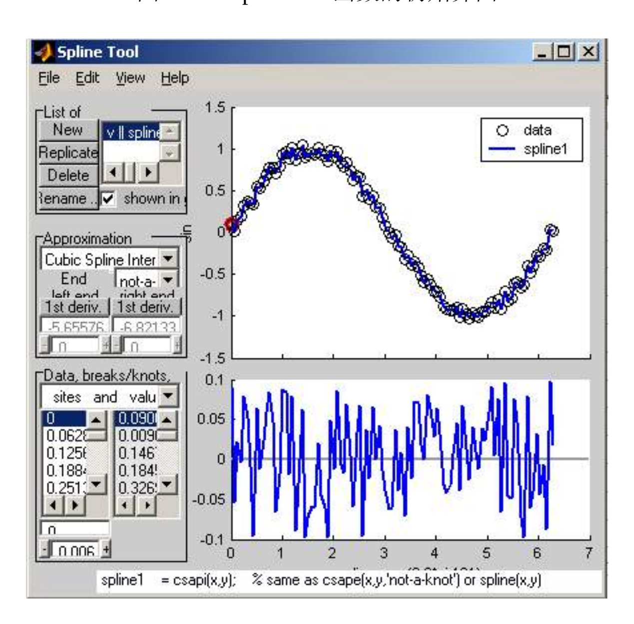


图 38-4　splinetool 函数的 GUI 界面

表 38-1　splinetool 函数 GUI 界面上 approximation 项的可选参数

生成样条曲线的方法	可 调 情 况
三次样条插值法(Cubic Spline Interpolation)	可选择各种约束条件
平滑样条法(Smoothing Spline)	可按 4 阶或 6 阶（函数的阶次）进行样条平滑处理；可改变精度的允许值；可调整鲁棒性分析时的权重
最小二乘拟合法 (Least-Squares Approximation)	函数的阶次可从 1 到 14 变化，默认值是 4，即三次样条插值；可修改分段多项式的段数；可增加或减少数据点以改变精度；可调整鲁棒性分析时的权重
样条插值法(Spline Interpolation)	函数的阶次可从 1 到 14 变化，默认值是 4，即三次样条插值。如果提供的默认数据点数组拟合生成的样条曲线不是非常好，可以移动数据点以使拟合效果更好

在图形用户界面下，可以使用同一批数据，产生与上述方法相对应的样条曲线，并对各种曲线图进行比较。另外，从界面上可以得到以下信息：

（1）当前每个数据点的坐标值（x, y）。

（2）可供选择的方法。

（3）当前选择的方法的辅助图形。可以从菜单 view 中，选择产生样条函数的一阶、二阶导数曲线图以及误差曲线图，默认为误差曲线图。

在图形用户界面下，可以给图形作图例，并且选择 File 菜单下的 Print 子菜单，把图形打印出来。也可以选择 File 菜单下的 Export Data 与 Export Spline 子菜单，将数据与样条函数输出到工作空间(workspace)，以便进一步分析。

在 Edit 菜单下，可以选择 add data，delete data 或 move data 子菜单，来增加、删除或移动数据点。在图形窗口内，右击鼠标也能弹出上述子菜单。

【例 38-1】 用三次样条插值法生成样条曲线

本例子演示在各种约束条件下插值生成三次样条曲线的情况。在函数 splinetool 初始界面中任意选择一类数据，例如选择 Provide your own data，然后在随后出来的界面中接受默认值，就会显示图 38-5 所示的界面。

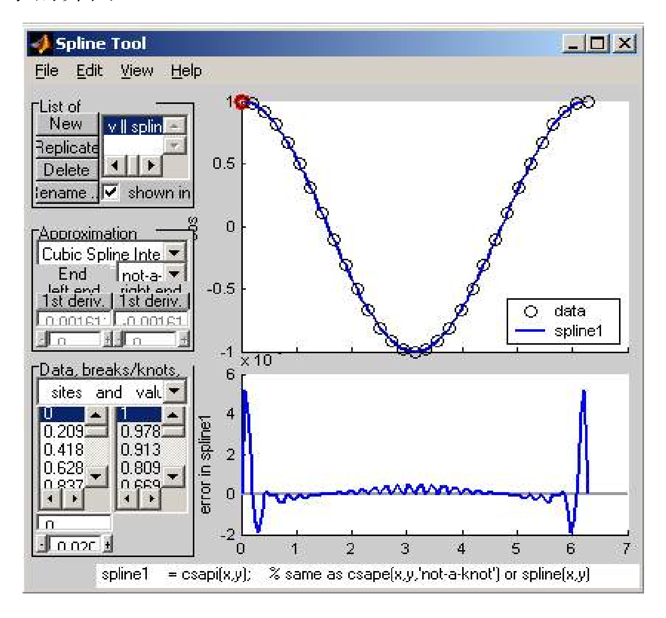


图 38-5 插值生成的三次样条曲线 GUI 界面

默认的生成样条曲线的方法就是在 not-a-knot 约束条件下的三次样条插值法(Cubic Spline Interpolation)。它生成 x= linspace(0, 2*pi, 21)，y=cos(x)的样条曲线。注意图中的误差曲线，相对误差的最大值在两端点附近，约为 5e-6。

下面对在各种约束条件下生成三次样条曲线的情况进行对比。

（1）从约束条件组合框中，选择约束条件 complete。因为余弦函数 cos 的一阶导数是正弦函数 sin，因此，左右端点的一阶导数值是零。注意图中的误差曲线，相对误差的最大值在两端点附近，约为 5e-6。由界面底下的 Bottomline 栏可知，此样条曲线由函数 csape 生成，图形如 38-6 所示。

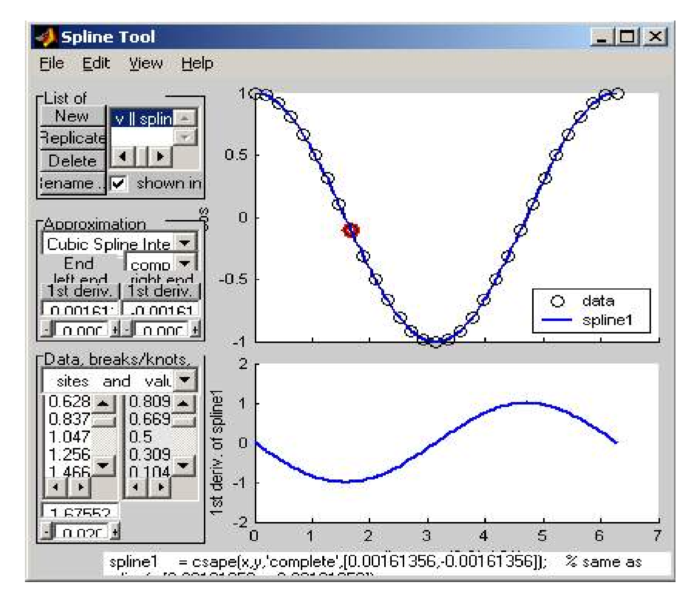


图 38-6 cos 函数在 complete 约束条件下插值生成的三次样条曲线 GUI 界面

（2）从约束条件组合框中，选择约束条件 natural。从图中的误差曲线可知，误差变大，其相对误差最大值约为 2e-3，图形如 38-7 所示。

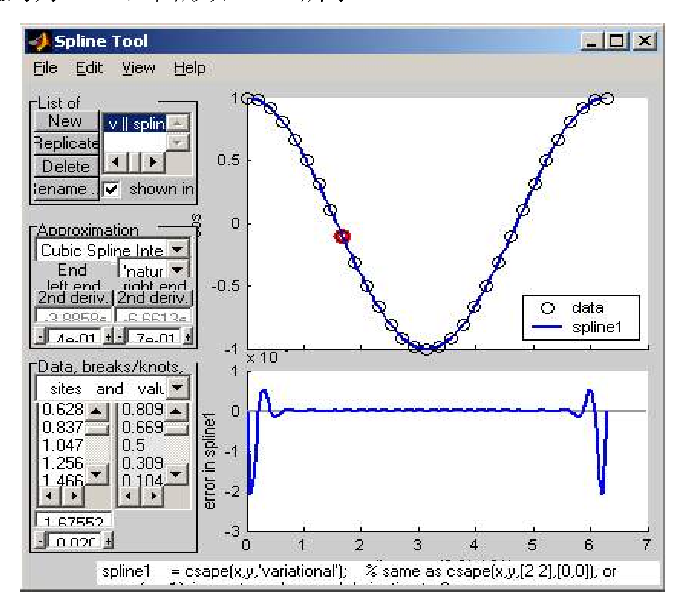


图 38-7 cos 函数在 natural 约束条件下插值生成的三次样条曲线 GUI 界面

（3）从约束条件组合框中，选择约束条件 periodic，误差又变小了，其相对误差最大值约为 5e-6。

【例 38-2】 用平滑样条法生成样条曲线

在图形用户界面上，选择 File 主菜单下的 Restart 子菜单，重新开始选择数据。在显示的 splinetool 初始界面中选择 Census data，然后，再选择平滑样条法(Smoothing Spline Interpolation)生成样条曲线。图形如 38-8 所示。

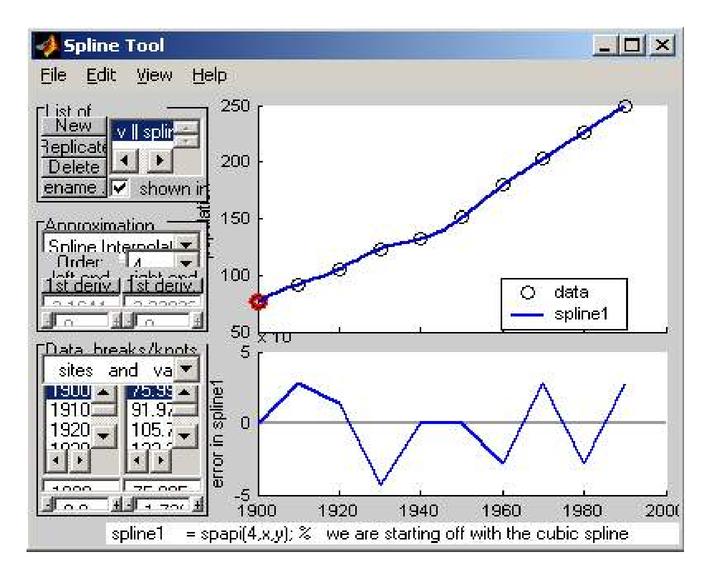


图 38-8　平滑样条法生成的样条曲线 GUI 界面

从误差曲线图可知，用平滑样条法生成的样条曲线误差很大。实际上，对于 Census data 数据，采用三次样条插值法生成的样条曲线，其误差就很小，图 38-9 就是在约束条件 not-a-knot 下利用三次样条插值法生成的样条曲线，请注意其误差曲线。

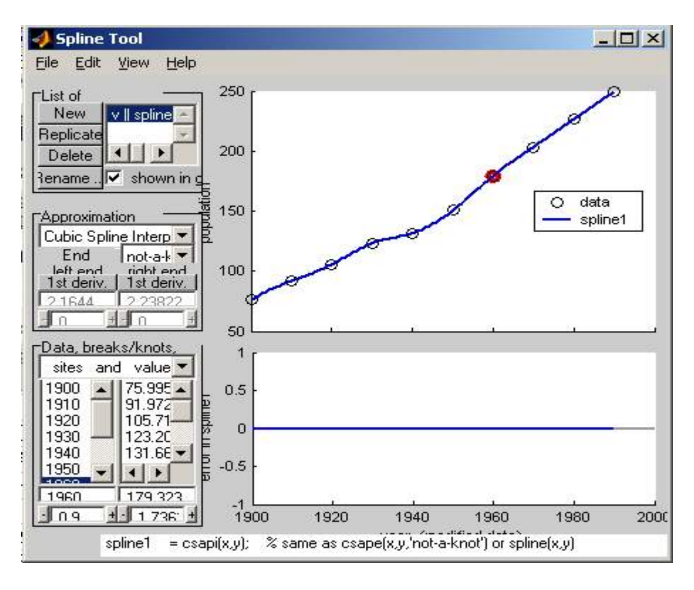


图 38-9　三次样条插值法生成的样条曲线 GUI 界面

【例 38-3】　用最小二乘拟合法生成样条曲线。

在上面的图形用户界面上，选择 File 主菜单下的 Restart 子菜单，重新开始选择数据。在显示的 splinetool 初始界面中选择 Titanium heat data，然后，再选择最小二乘拟合法(Least-Squares Approximation)生成样条曲线，图形如 38-10 所示。

由于中间的数据点非常少，因此用最小二乘法拟合时，生成的样条曲线的误差非常大。数据点的(x, y)坐标值在数据点坐标值列表框中(Data, breaks/knots, weights…)。注意，两个端节点都是 4

重。单击鼠标右键，从弹出的菜单中选择增加数据点菜单(Add Points…)，然后在误差最大处增加一个节点。此时，由于中间多了一个数据点，误差就明显地变小了，如图 38-11 所示。

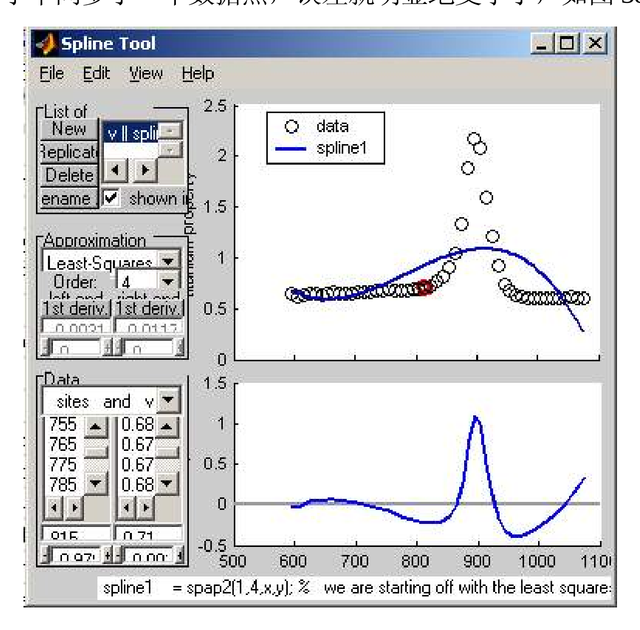


图 38-10　最小二乘法生成的样条曲线 GUI 界面

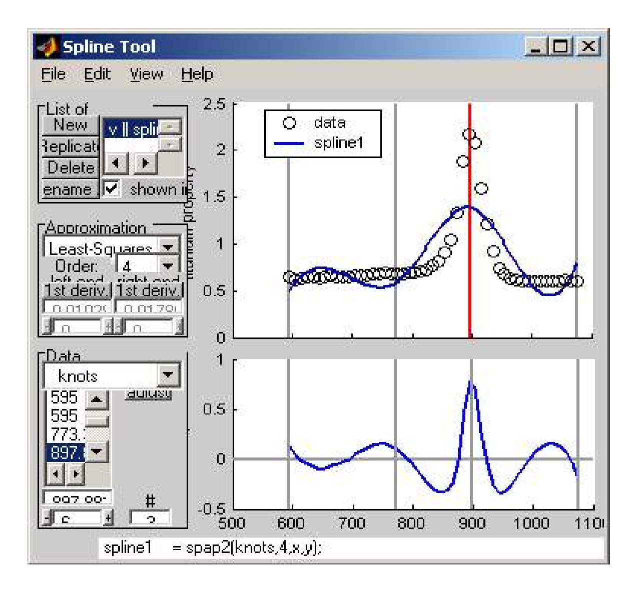


图 38-11　增加一个节点后最小二乘法生成的样条曲线 GUI 界面

单击鼠标右键，从弹出的菜单中选择复制数据点菜单(Copy Points)，将增加当前数据点的重次。当然，也可以选择删除数据点菜单(Delete Points)，删除当前选择的数据点。

在 Data, breaks/knots, weights 列表框的# pieces 编辑框内，输入分段多项式的段数，如输入 6，然后单击调整(Adjust)按纽，就把当前区间划分为 6 段，如图 38-12 所示。

【例 38-4】　平滑样条法。

在 splinetool 函数的 GUI 界面的 Approximation 中，选择 smoothing spline，就用平滑样条法生成样条曲线。生成的图形如 38-13 所示。

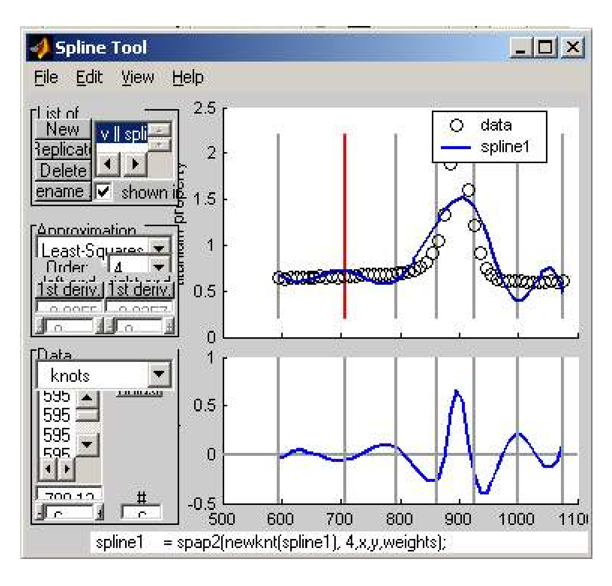


图 38-12　分为六段后最小二乘法生成的样条曲线 GUI 界面

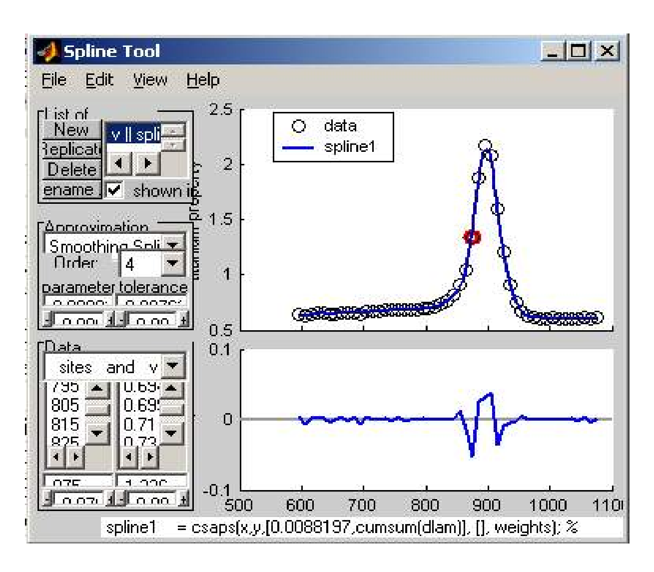


图 38-13　平滑样条法生成的样条曲线 GUI 界面

当平滑参数 parameter 从 0 到 1 变化时，平滑方法从线性最小二乘法变到在约束条件 natural 下的三次样条插值法。当允许的精度系数 tolerance 从 0 变到很大值，最大甚至为 inf（无穷大），平滑方法从最好的在约束条件 natural 下的三次样条插值法变到线性最小二乘法。即增大平滑参数 parameter 值或减小允许的精度系数 tolerance 值，误差值都减少。然而，误差变小却使鲁棒性变大，这从二阶导数曲线中可以得知。从 view 菜单下选择 Show 2nd Derivative 子菜单，就可以看到二阶导数曲线。

可以通过修改误差权重使样条曲线通过特殊点；也可以通过修改鲁棒权重使二阶导数曲线更光滑。

第五篇　信号处理工具箱

第 39 章　采样与波形发生

数字信号处理的对象，是在采样时种的控制下，通过 A/D 转换器以一定的采样率对模拟信号进行采样而得到的。由采样定理可知，采样频率必须大于模拟信号最高频率的 2 倍。然而在许多情况下，要求信号以不同的频率采样。这时需要对采样数据进行处理。利用抽取或内插方法改变采样率。

有关函数如下。

1．upfirdn 函数

upfirdn 函数改变信号的采样率（用于 FIR 滤波器），其调用格式为：

● upfirdn(X, H, P, Q)　返回信号 X 通过 3 个级联系统后的输出结果。

（1）过采样（插入零值使采样频率上升为原来的 P 倍）。

（2）滤波（H 给定的单位响应滤波器）。

（3）欠采样（通过抽取采样点使采样率下降 Q 倍）。

当函数调用形式为 upfirdn(X, H, P)时，Q 默认为 1。

当函数调用形式为 upfirdn(X, H)时，P，Q 默认为 1，此时函数返回 X 经一冲激响应为 H 的 FIR 滤波器滤波后的结果。

【例 39-1】　有一信号的采样信号 sig，其最高频率分量为 10000Hz，采样频率为 40000Hz，采样点数为 1024，试将采样频率降为 4000Hz。

解：h=fir1(112, 0.1)；%抗混叠滤波，将采样频率的一半 2000Hz 以上的%2000/20000=0.1。
　　sig1=upfirdn(sig,h,1,40000/4000)；

2．decimate 函数

decimate 函数对信号欠采样（先经低通滤波），其调用格式为：

● Y=decimate（X, R）　返回的是向量 X 的重采样系列，采样率为原来的 1/R 倍。

【例 39-2】　有一信号的采样信号 sig，其最高频率分量为 10000Hz，采样频率为 40000Hz，采样点数为 8000，试将采样频率降为 400Hz。

解：　sig1=upfirdn(sig, 10)；
　　　sig2=upfirdn(sig1, 10)；

decimate 使用的是一 8 阶的 Chebyshev I 型低通滤波器。当 R 很大时，设计的 Chebyshev 滤波器由于数字精度的限制可能不准确。在此情况下，decimate 将使用一个阶数较低的滤波器。为了得到较好的抗混叠性能，可将 R 分解为它的因子，然后分几次调用 decimate 函数。

3．interp 函数

interp 函数对信号进行内插，内插增加了系列的采样率，即在原始系列中插入零值，其调用格式为：

- y = interp(x, r) 增加 x 的采样率为原来的 r 倍。插入后的向量 y 是输入向量 x 的 r 倍。
- y = interp(x, r, l,alpha) 限定了滤波器的长度为 l，截止频率为 alpha，l 的默认值为 4, alpha 的默认值为 0.5。
- [y, b] = interp(x, r, l, alpha) 返回向量 b 为插入所用的滤波器系数向量。

【例 39-3】 对信号 $x = \sin(2\pi\times30t) + \sin(2\pi\times60t)$ 进行 4 倍内插。

程序如下：

```
%program  p101
t = 0:0.001:1; % Time vector
x = sin(2*pi*30*t) + sin(2*pi*60*t);
y = interp(x,4);
subplot(121)
stem(x(1:30));
title('Original Signal');
subplot(122)
stem(y(1:120));
title('Interpolated Signal');
```

运行结果如图 39-1 所示。

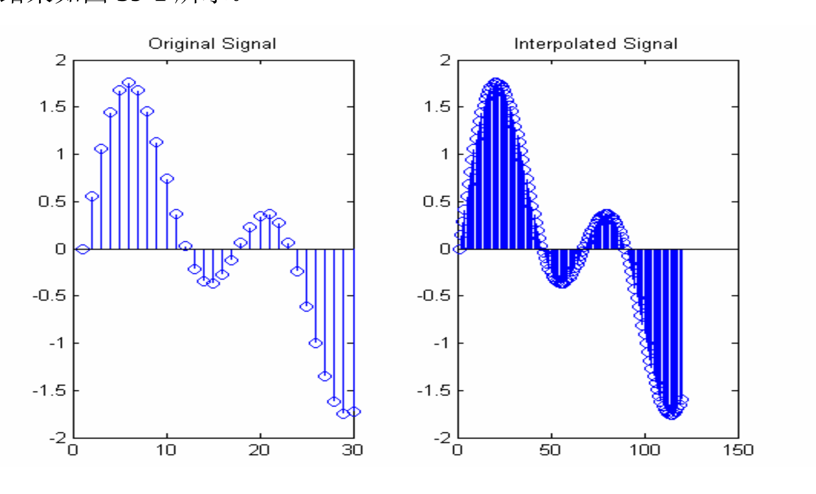


图 39-1　原信号和内插后的信号

4．resample 函数

该函数调用格式为：

- y = resample(x, p, q)
- y = resample(x, p, q, n)
- y = resample(x, p, q, n, beta)
- y = resample(x, p, q, b)
- [y, b] = resample(x, p, q)

函数以 p/q 倍速度对向量 x 进行重采样。

【例 39-4】 以原采样率的 3/2 重采样线性系列 x。

```
%program p102
fs1 = 10;              % Original sampling frequency in Hz
t1 = 0:1/fs1:1;        % Time vector
x = t1;                % Define a linear sequence
y = resample(x,3,2);   % Now resample it
t2 = (0:(length(y)-1))*2/(3*fs1); % New time vector
plot(t1,x,'*',t2,y,'o',-0.5:0.01:1.5,-0.5:0.01:1.5,':')
legend('original','resampled');
xlabel('Time')
```

运行结果如图 39-2 所示。

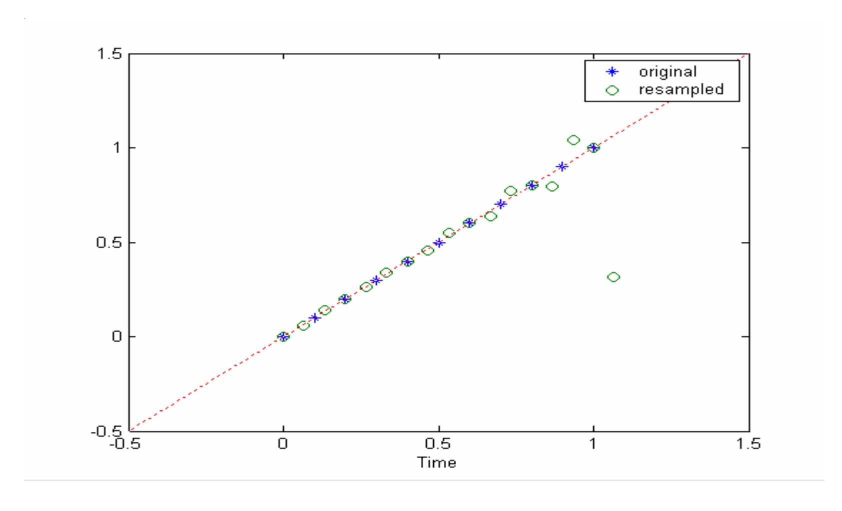


图 39-2 重采样信号

5. diric 周期函数（Dirichlet 函数）

该函数的调用格式为：

● Y=diric（X, N）返回一大小与 X 相同的矩阵，其元素为 Dirichlet 函数。N 必须为正。该函数将 0 到 2*π 等间隔分成 N 等份。

Dirichlet 函数定义为

$$\text{diric} = \begin{cases} -1^{\frac{x}{2\pi}(n-1)} & x = 0, \pm 2\pi, \pm 4\pi \ldots \\ \dfrac{\sin(x/2)}{n\sin(x/2)} & \text{else} \end{cases}$$

【例 39-5】 绘出 diric 函数的图形。

```
% program  p103
x = linspace(0,4*pi,300);
subplot(121)
plot(x,diric(x,7))
subplot(122)
plot(x,diric(x,8))
```

运行结果如图 39-3 所示。

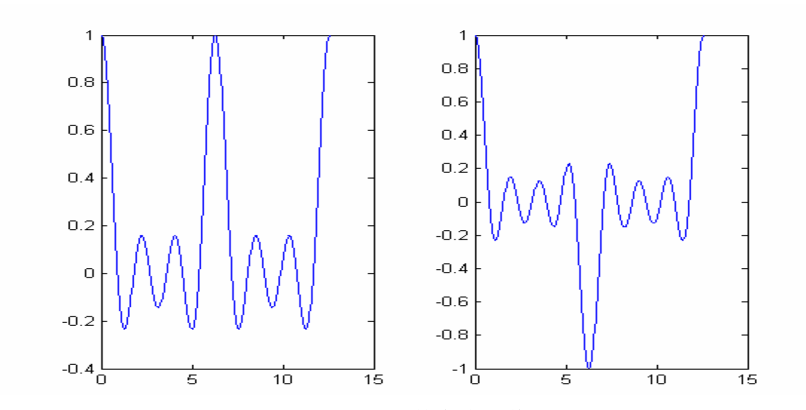


图 39-3　diric 函数产生的信号图

6．gausplus 函数

gausplus 函数为高斯函数调幅的正弦波发生器，其调用格式为：

- YI=gauspuls(T, Fc, BW)　返回最大幅度为 1 的高斯函数调幅的正弦波的采样，其中心频率为 Fc（Hz），相对带宽为 BW，时间由数组 T 给定。注意：BW 必须大于 0。默认时，Fc=1 000Hz, BW=0.5。
- YI=gauspuls(T, Fc, BW, BWR)
- [YI, YQ]=gauspuls(…)
- [YI, YQ, YE]=gauspuls(…)
- TC=gauspuls(‘cutoff’, FC, BW, BWR, TPE)

【例 39-6】　绘出一 50 kHz 高斯 RF 脉冲，相对带宽为 60%，采样速率 1 MHz. 包络下降为峰值的 40 dB 以下。

程序如下：

```
%program p104
tc = gauspuls('cutoff',50e3,0.6,[],-40);
t = -tc : 1e-6 : tc;
yi = gauspuls(t,50e3,0.6);
plot(t,yi)
```

运行结果如图 39-4 所示。

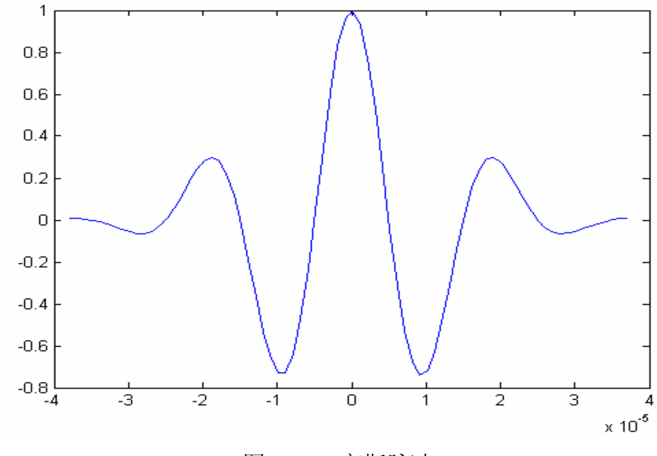


图 39-4　高斯脉冲

7．sawtooth 函数

该函数调用格式为：

- x = sawtooth(t)
- x = sawtooth(t, width)

锯齿波函数产生一锯齿波，其峰值为±1，周期为 2π。可选的宽度参数限定了信号的最大值出现的位置。

【例 39-7】 产生 1.5s 的 50 Hz 的锯齿波,采样率为 10 kHz，画出 0~0.2s 内的波形。

程序如下：

```
%program p105
fs = 10000;
t = 0:1/fs:1.5;
x = sawtooth(2*pi*50*t);
plot(t,x),
axis([0 0.2 -1 1])
```

运行结果如图 39-5 所示。

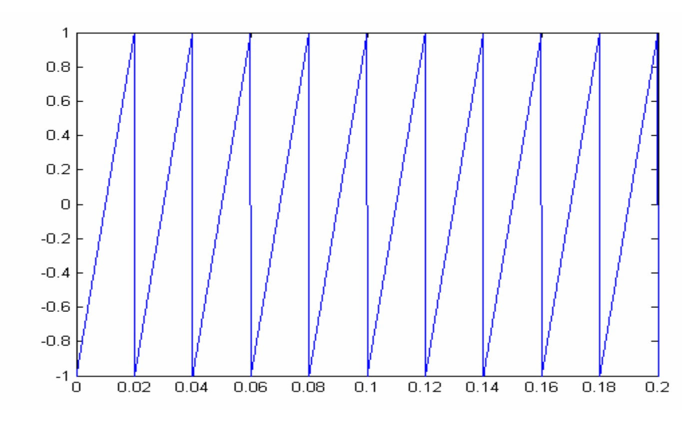


图 39-5　锯齿波

8．sinc 函数

sinc 函数定义为：

$$\sin c(t)\begin{cases}1, & t=0\\ \dfrac{\sin(\pi t)}{\pi t}, & t\neq 0\end{cases}$$

它是宽度为 2π，高为 1 的矩形脉冲的连续傅立叶反变换。

$$\sin c(t)\left\{\frac{1}{2\pi}\int_{-\pi}^{\pi}\mathrm{e}^{j\omega t}\mathrm{d}\omega\right.$$

调用格式为：

- y = sinc(x)，函数返回与 x 相同大小的向量 y。

【例 39-8】 绘出 sinc 函数，范围从-5 到 5。

```
%program p106
x = linspace(-5,5);
y = sinc(x);
```

```
plot(x,y)
```

运行结果如图 39-6 所示。

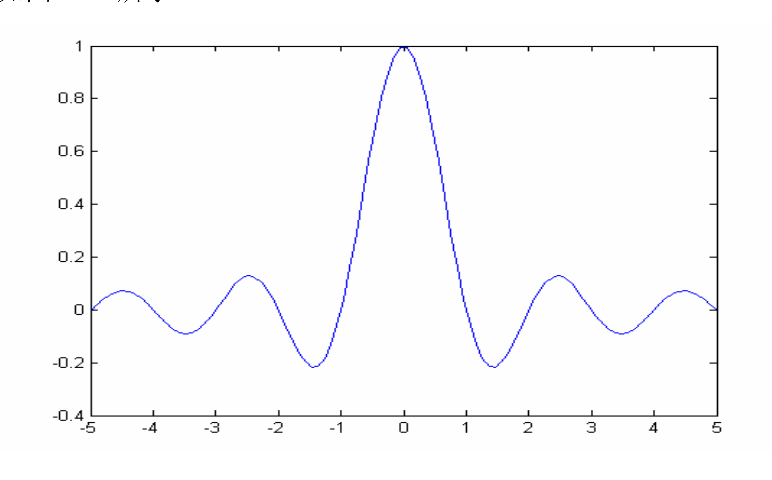


图 39-6 sinc 函数图

9. square 函数

该函数调用格式为：

- x = square(t)
- x = square(t, duty)

square（方波）函数产生一周期为 2π 的方波，参数 duty 限定了信号为正的百分比。

【例 39-9】 产生一方波，正信号占 40%。

程序如下：

```
%program p107
t = 0:0.01:6*pi;
x = square(t,40);
plot(t,x),
axis([0 6*pi -1 1.5])
```

运行结果如图 39-7 所示。

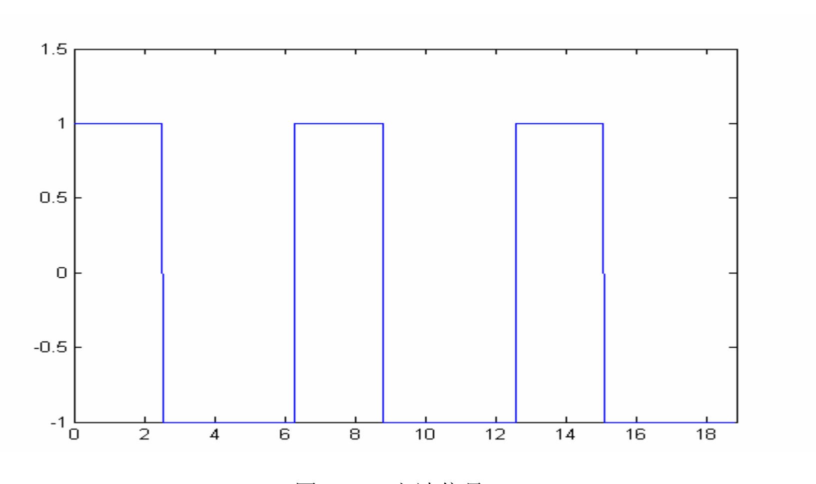


图 39-7 方波信号

第 40 章　模拟滤波器设计

滤波器是一个选频装置。理想的滤波器应能无失真地通过有用信号，而使无用信号完全抑制。理想滤波器频率特性可以描述为

$$H(j\omega)=\begin{cases}k\mathrm{e}^{j\omega t_d}, & 在通带内\\ 0\ , & 在阻带内\end{cases}$$

但理想的滤波器是物理上不可实现的系统。实际滤波器的频率特性只能“逼近”理想滤波器。通常设计要求的滤波器的技术指标包括通带波纹 R_P (Passband ripple)(dB)、阻带衰减 R_S (Stopband attenuation)(dB)、通带边界频率 ω_S 、过渡带宽（ $\omega_S-\omega_P$ ）。模拟滤波器的技术指标可由平方幅值响应函数 $A(\omega^2)=|H(j\omega)|^2$ 形式给出，而 $A(\omega^2)$ 和传递函数的关系如下：

$$A(\omega^2)=|H(j\omega)|^2=H(S)H(-S)\big|_{S=j\omega}$$

即

$$A(\omega^2)\big|_{\omega^2=-S^2}=H(S)H(-S)$$

所以，当给定了模拟滤波的技术指标后，由关系式可以求出 $A(-S^2)$ 和 $H(S)$ 。但是 $H(S)$ 的极点必须落在 S 平面左半平面，滤波器才会稳定。要使 $H(S)$ 具有最小相位时，零点也应选在左半平面。

下面将对模拟原型滤波器的特点及设计方法进行分析。

40.1　巴特沃思滤波器

巴特沃思模拟低通滤波器原型的平方幅值响应函数为：

$$|H(j\omega)|^2=A(\omega^2)=\frac{1}{1+(\dfrac{\omega}{\omega_C})^{2N}}$$

式中， ω_C 为低通滤波器的截止频率，N 为滤波器的阶数。

Butterworth 低通滤波器的特点是：通带内具有最平坦的幅频特性，且随频率增大平滑单调下降；阶数 N 越高，特性越接近矩形，过渡带越窄。传递函数无零点，极点等距离分布在以 $|S|=\omega_C$ 为半径的圆周上。

40.1.1　有关函数介绍

MATLAB 信号处理工具箱提供的 Butterworth 模拟低通滤波器原型函数为：

- [Z, P, K] = buttap(n)

其中，n 为 Butterworth 滤波器阶数；Z，P，K 为滤波器零点、极点和增益。函数返回 n 阶 Butterworth 模拟低通滤波器原型的极点和增益。因为无零点，所以 Z 是空矩阵。

40.1.2　应用实例

【例 40-1】　绘制 10 阶 Butterworth 低通滤波器的平方幅频响应曲线。

程序如下：

```
%MATLAB PROGRAM p201
%Butterworth analog lowpass prototype
n=0:0.01:2;
N=10
[z,p,k]=buttap(N);
[b,a]=zp2tf(z,p,k);%Zero-pole to transfer function conversion.
[H,w]=freqs(b,a,n);%returns the complex frequency response vector H of the
filter b/a:
magH=(abs(H)).^2;
plot(w,magH);
axis([0 2 0 1]);%AXIS([XMIN XMAX YMIN YMAX]) sets scaling for the x- and
y-axes on the
%current plot
xlabel('w/wc');
ylabel('|H(jw)|^2');
title('Butterworth analog filter prototype');
grid
```

程序运行结果如图 40-1 所示。

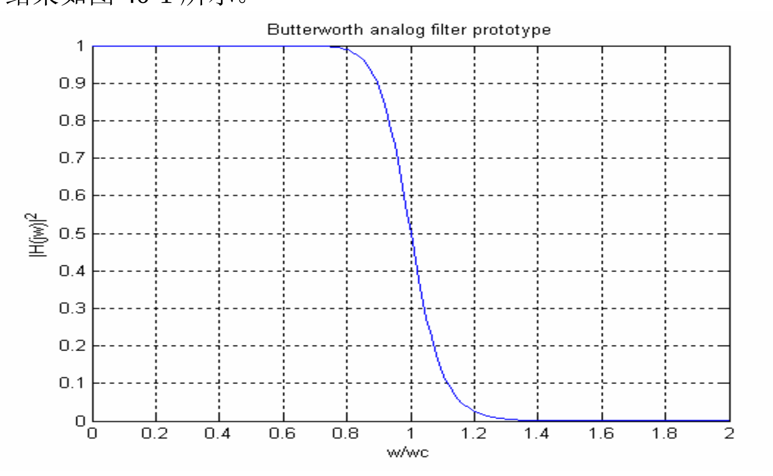


图 40-1　Butterworth 低通滤波器原型平方幅频响应曲线

40.2　切比雪夫滤波器

切比雪夫（Chebyshev）滤波器有 I 型和 II 型两类。

40.2.1　Chebyshev I 型

MATLAB 函数：

- [Z，P，K] = cheb1ap(N,Rp)

N 为阶数；Rp 为通带波纹（dB）；Z，P，K 为滤波器零点、极点和增益。

滤波器的平方幅值响应函数为

$$|H(j\omega)|^2 = A(\omega^2) = \frac{1}{1+\varepsilon^2 C_N^2(\dfrac{\omega}{\omega_C})}$$

式中，ε 为小于 1 的正数，表示通带内幅值波纹情况；ω_C 为截止频率，N 为多项式 $C_N(\dfrac{\omega}{\omega_C})$ 的阶数。

$$C_N(x) = \begin{cases} \cos(N\cos^{-1}(x)), & |x| \leqslant 1 \\ \cosh(N\cosh^{-1}(x)), & x \leqslant 1 \end{cases}$$

Chebyshev I 型滤波器的特点：通带内具有等波纹起伏特性，阻带内单调下降且衰减更大，随着 N 的增大接近矩形。传递函数无零点，极点分布在椭圆上。

【例 40-2】 绘出 8 阶 Chebyshev I 型模拟低通滤波器的平方幅频响应曲线。

程序如下：

```
%PROGRAM p202
%chebyshev type I analog lowpass prototype
n=0:0.01:2;
N=8;
Rp=1;
[z,p,k]=cheb1ap(N,Rp);
[b,a]=zp2tf(z,p,k);%Zero-pole to transfer function conversion.
[H,w]=freqs(b,a,n);%returns the complex frequency response vector H of the
      filter b/a:
magH=(abs(H)).^2;
plot(w,magH);
axis([0 2 0 1]);%AXIS([XMIN XMAX YMIN YMAX]) sets scaling for the x- and
y-axes on
%the current plot
xlabel('w/wc');
ylabel('|H(jw)|^2');
title('N=8');
grid
```

运行结果如图 40-2 所示。

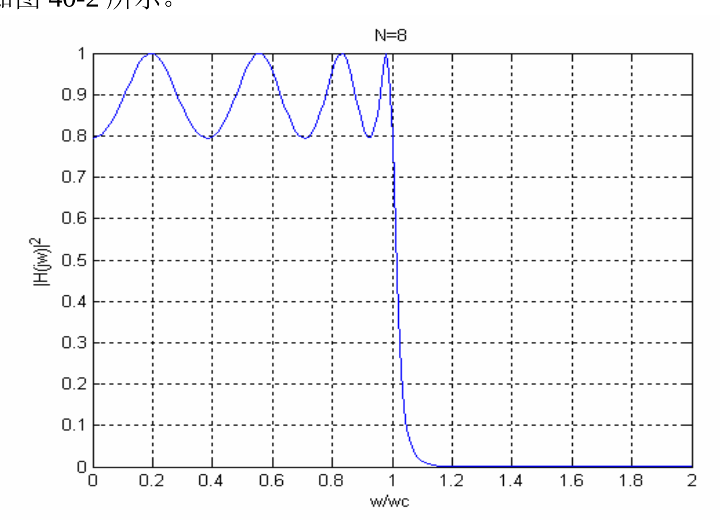


图 40-2 Chebyshev I 低通滤波器原型平方幅频响应曲线

40.2.2 Chebyshev II 型

MATLAB 函数：

- [Z, P, K] = cheb2ap(N, Rs)

N 为阶数，Rs 为阻带波纹（dB）, Z, P, K 为滤波器零点、极点和增益。

滤波器的平方幅值响应函数为

$$\left|H(j\omega)\right|^2 = A(\omega^2) = \frac{1}{\left[1+\varepsilon^2 C_N^2(\frac{\omega}{\omega_C})\right]^{-1}}$$

式中，ε为小于 1 的正数，表示阻带内幅值波纹情况；ω_C为截止频率，N为多项式$C_N(\frac{\omega}{\omega_C})$的阶数。

$$C_N(x)=\begin{cases}\cos(N\cos^{-1}(x)), & |x|\leqslant 1\\ \cosh(N\cosh^{-1}(x)), & x>1\end{cases}$$

Chebyshev II 型滤波器的特点：阻带内具有等波纹起伏特性，通带内单调、平滑，随着 N 的增大接近矩形。传递函数有零点，极点。

【例 40-3】 绘出 8 阶 Chebyshev II 型模拟低通滤波器的平方幅频响应曲线。

程序如下：

```
%PROGRAM p203
%chebyshev type II analog lowpass prototype
n=0:0.01:2;
N=8;
Rs=10;
[z,p,k]=cheb2ap(N,Rs);
[b,a]=zp2tf(z,p,k);%Zero-pole to transfer function conversion.
[H,w]=freqs(b,a,n);%returns the complex frequency response vector H of the
      filter b/a:
magH=(abs(H)).^2;
plot(w,magH);
axis([0 2 0 1.1]);%AXIS([XMIN XMAX YMIN YMAX]) sets scaling for the x- and y-axes
%on the current plot
xlabel('w/wc');
ylabel('|H(jw)|^2');
title('N=8');
grid
```

程序运行结果如图 40-3 所示。

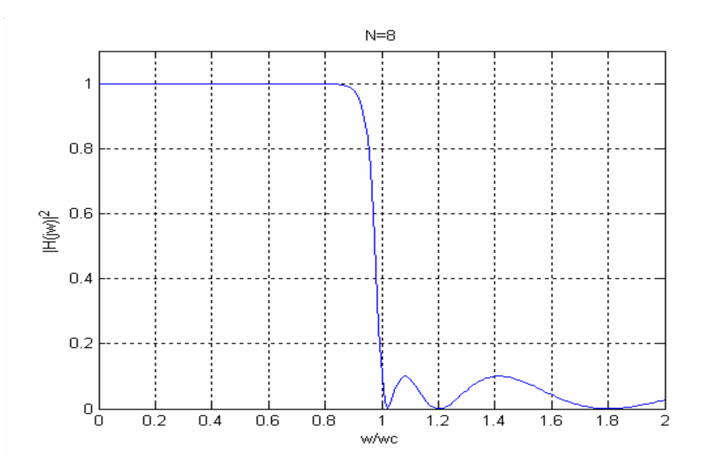


图 40-3　Chebyshev II 低通滤波器原型平方幅频响应曲线

40.3　椭圆滤波器

40.3.1　有关函数介绍

该函数的调用格式为：

- [Z, P, K] = ellipap (N, Rp, Rs)

N 为阶数，Rp 为通带波纹（dB），Rs 为阻带波纹（dB），Z, P, K 为滤波器零点、极点和增益。

滤波器的平方幅值响应函数为

$$|H(j\omega)|^2 = A(\omega^2) = \frac{1}{1+\mu^2 E_N^2(\dfrac{\omega}{\omega_C})}$$

式中，μ 为小于 1 的正数，表示波纹情况，ω_C 为截止频率，N 为多项式 $E_N(\dfrac{\omega}{\omega_C})$ 的阶数。

椭圆滤波器特点是通带和阻带内都具有等波纹起伏特性，但相频响应具有明显非线性。

40.3.2　应用实例

【例 40-4】　绘出 8 阶椭圆滤波器的平方幅频响应曲线。

程序如下：

```
%PROGRAM p204
%Elliptic analog lowpass prototype
n=0:0.01:2;
N=8;
Rp=1;
Rs=10;
[z,p,k]=ellipap(N,Rp,Rs);
[b,a]=zp2tf(z,p,k);%Zero-pole to transfer function conversion.
[H,w]=freqs(b,a,n);%returns the complex frequency response vector H of the
filter b/a:
magH=(abs(H)).^2;
```

```
plot(w,magH);
axis([0 2 0 1]);%AXIS([XMIN XMAX YMIN YMAX]) sets scaling for the x- and
y-axes on the
%current plot
xlabel('w/wc');
ylabel('|H(jw)|^2');
title('N=8');
grid
```

程序运行结果如图 40-4 所示。

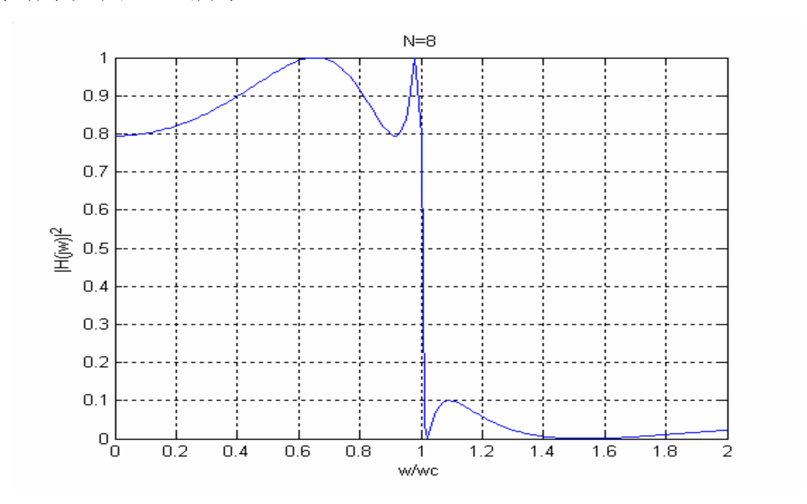


图 40-4　椭圆低通滤波器原型平方幅频响应曲线

40.4　贝塞尔滤波器

40.4.1　有关函数介绍

MATLAB 设计函数为：

- [Z, P, K] = besselap (N)

N 为阶数，小于 25; Z, P, K 为滤波器零点、极点和增益。

贝塞尔滤波器的特点：在零频时具有最平坦的群延迟，并在整个通带内群延迟几乎不变。所以贝塞尔滤波器在通带内保持形状不变。

40.4.2　应用实例

【例 40-5】　绘出 10 阶贝塞尔滤波器的平方幅频和相频响应曲线。

程序如下：

```
%PROGRAM p205
%bessel lowpass prototype
n=0:0.01:2;
N=10;
[z,p,k]=besselap(N);
[b,a]=zp2tf(z,p,k);%Zero-pole to transfer function conversion.
```

```
[H,w]=freqs(b,a,n);%returns the complex frequency response vector H of the
       filter b/a
magH2=(abs(H)).^2;
phaH=unwrap(angle(H));
phaH=phaH*180/pi;
subplot(211);
plot(w,magH);
axis([0 2 0 1]);%AXIS([XMIN XMAX YMIN YMAX]) sets scaling for the x- and
      y-axes on the
%current plot
xlabel('w/wc');
ylabel('|H(jw)|^2');
title(['N=',num2str(N)]);
grid
%display phrase_freq figure
subplot(212)
plot(w,phaH);
xlabel('w/wc');
ylabel('phase');
grid
```

程序运行结果如图 40-5 所示。从图可以看出相频曲线几乎为线性。

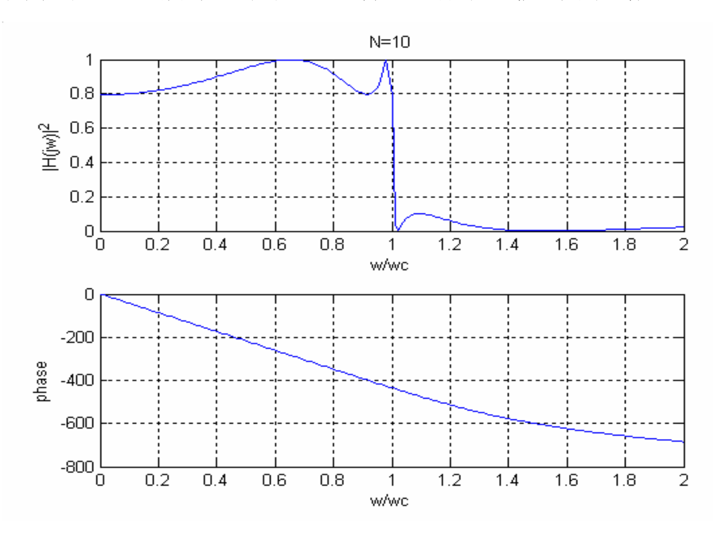


图 40-5　bessel 低通滤波器原型的幅频和相频响应曲线

40.5　频率变换

通过传递函数中频率自变量之间的变换关系，可以从模拟低通滤波器原型获得模拟的低通滤波器、高通滤波器、带通滤波器和带阻滤波器，再经过 S 域至 Z 域转换又可以设计数字 IIR 滤波器。

40.5.1　有关函数介绍

频率变换函数有 lp2lp, lp2hp, lp2bp 和 lp2bs 4 个，下面将分别加以分析。

1．lp2lp 函数

lp2lp 函数用于实现低通模拟原型滤波器至低通滤波器的频率变换。即

$$H(s)=H(p)\big|_{p=\omega_0 s}$$

式中，$H(p)$ 为低通模拟原型滤波器传递函数，$H(s)$ 为低通滤波器传递函数。

该函数的调用格式为：

- [bt, at] = lp2lp(b, a, Wo)
- [At, Bt, Ct, Dt] = lp2lp(A, B, C, D, Wo)

Wo 为低通滤波器期望截止频率(rad/sec)。

2．lp2hp 函数

lp2hp 函数用于实现低通模拟原型滤波器至高通滤波器的频率变换。即

$$H(s)=H(p)\Big|_{p=\frac{\omega_0}{s}}$$

式中，$H(p)$ 为低通模拟原型滤波器传递函数，$H(s)$ 为高通滤波器传递函数。

调用格式为：

- [bt, at] = lp2hp(b, a, Wo)
- [At, Bt, Ct, Dt] = lp2hp(A, B, C, D, Wo)

Wo 为高通滤波器期望截止频率(rad/sec)。

3．lp2bp 函数

lp2bp 函数用于实现低通模拟原型滤波器至带通滤波器的频率变换。即

$$H(s)=H(p)\Bigg|_{p=\frac{\omega_0}{B_w}\frac{(\frac{s}{\omega_0})^2+1}{\frac{s}{\omega_0}}}$$

式中，$H(p)$ 为低通模拟原型滤波器传递函数，$H(s)$ 为带通滤波器传递函数。

调用格式为：

- [bt, at] = lp2bp(b, a, Wo, Bw)
- [At, Bt, Ct, Dt] = lp2bp(A, B, C, D, Wo, Bw)

Wo 为带通滤波器期望截止频率(rad/sec)，Bw 为带通滤波器带宽(rad/sec)。

4．lp2bs 函数

lp2bs 函数用于实现低通模拟原型滤波器至带阻滤波器的频率变换。即

$$H(s)=H(p)\Bigg|_{p=\frac{\omega_0}{B_w}\frac{\frac{s}{\omega_0}}{(\frac{s}{\omega_0})^2+1}}$$

式中，$H(p)$ 为低通模拟原型滤波器传递函数，$H(s)$ 为带阻滤波器传递函数。

调用格式为：

- [bt, at] = lp2bs(b, a, Wo, Bw)

● [At, Bt, Ct, Dt] = lp2bs(A, B, C, D, Wo, Bw)

Wo 为带阻滤波器期望截止频率(rad/s)，Bw 为带阻滤波器带宽(rad/s)。

40.5.2　应用实例

【例 40-6】　设计一个 9 阶 chebshev I 型高通滤波器，通带纹波 10dB，下边界频率 400π rad/s，绘出幅频响应图。

程序如下：

```
%program p206
%lp2hp chebshev I
N=9;
Rp=10;
w=400*pi;
[z,p,k]=cheb1ap(N,Rp);
[b,a]=zp2tf(z,p,k);
%design analog highpass filter
[bt,at]=lp2hp(b,a,w1);
[h,w]=freqs(bt,at);
semilogy(w/pi,abs(h));
axis([0 3000 1e-8 1]);
xlabel('Freq.(pi)');
grid on
```

程序运行结果如图 40-6 所示。

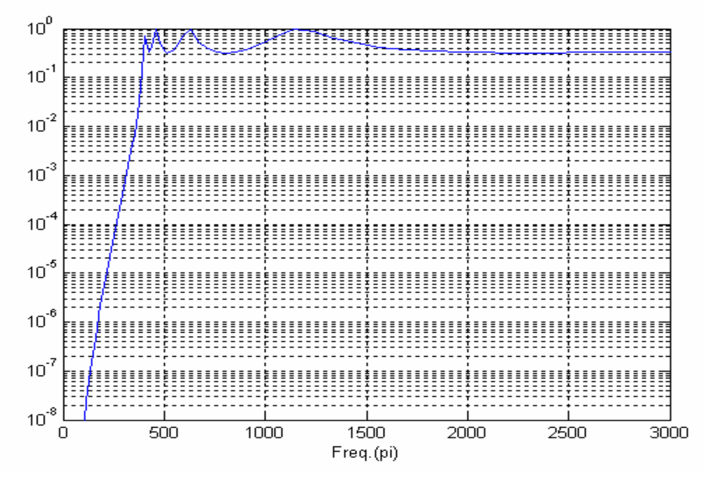


图 40-6　模拟带通滤波器幅频响应图

综上所述，利用频率变换设计模拟滤波器的步骤为：

（1）给出模拟滤波器的性能指标，如截止频率 ω_0 或上、下边界频率（ω_1,ω_2）、波纹特性及阻带衰减等。

（2）确定滤波器阶数。

（3）设计模拟原型滤波器。

（4）由频率变换设计模拟滤波器。

40.6　模拟滤波器最小阶数的选择

模拟滤波器设计中，很重要的问题是需要确定滤波器的阶数，因为它是决定滤波器的品质因素。阶数小不能满足性能指标，阶数太大则难以实现。在满足性能指标的前题下，阶数应该尽可能小，利于实现。阶数和滤波器性能间存在一定关系，通过此关系可以求出滤波器阶数。有两中方法可以确定滤波器最小阶数，一是通过常用的最小阶数确定原理，二是用 MATLAB 信号处理工具箱中的函数直接计算。

40.6.1　有关函数介绍

1．buttord

buttord 函数用于计算 Butterworth 滤波器的阶数和截止频率，其调用格式为：

- [n, Wn] = buttord(Wp, Ws, Rp, Rs)
- [n, Wn] = buttord(Wp, Ws, Rp, Rs, 's')

2．cheb1ord

cheb1ord 函数用于计算 Chebyshev I 滤波器的阶数，其调用格式为：

- [n, Wn] = cheb1ord(Wp, Ws, Rp, Rs)
- [n, Wn] = cheb1ord(Wp, Ws, Rp, Rs, 's')

3．cheb2ord

cheb2ord 函数用于计算 Chebyshev II 滤波器的阶数，其调用格式为：

- [n, Wn] = cheb2ord(Wp, Ws, Rp, Rs)
- [n, Wn] = cheb2ord(Wp, Ws, Rp, Rs, 's')

4．ellipord

该函数用于计算椭圆模拟滤波器的阶数，其调用格式为：

- [n, Wn] = ellipord(Wp, Ws, Rp, Rs)
- [n, Wn] = ellipord(Wp, Ws, Rp, Rs, 's')

4 个函数均返回滤波器最小阶数 n 和截止频率 Wn（3dB 频率）。其中，Wp 为通带边界频率（rad/sec）；Ws 为阻带边界频率（rad/s）；Rp 为通带波纹，即滤波器在通带 0~Wp 间的最大幅值损失；Rs 为阻带衰减，即滤波器幅值从通带至阻带下降的 dB 数，'s'表示模拟滤波器（默认时函数适用于数字滤波器）。

40.6.2　应用实例

【例 40-7】　设采样率为 1000 Hz，设计一低通滤波器，通带为 0～40Hz，通带波纹 R_p <=3dB，阻带为 150 Hz 到 Nyquist 频率 frequency (500 Hz)，阻带衰减 R_s >60dB 画出滤波器的频率响应。

程序如下：

```
%program p207
Wp = 40/500; Ws = 150/500;
[n,Wn] = buttord(Wp,Ws,3,60)
```

```
[b,a] = butter(n,Wn);
freqz(b,a,512,1000); title('n=5 Butterworth Lowpass Filter')
```

程序运行结果如下。滤波特性见及图 40-7。

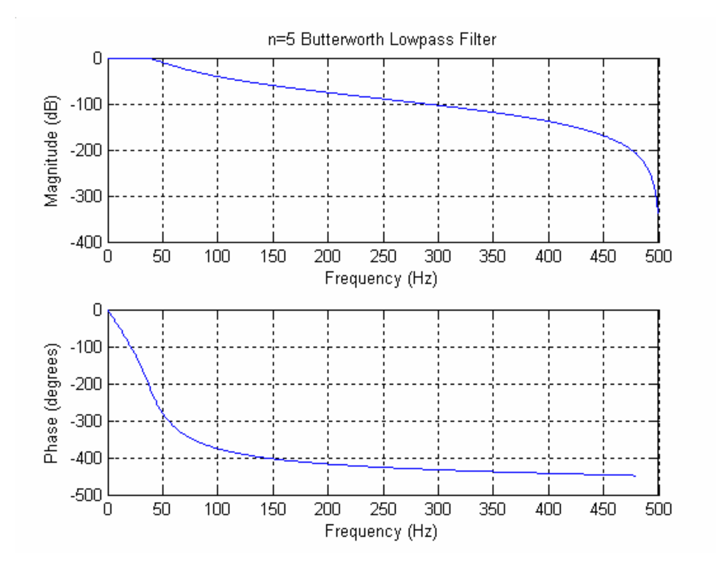


图 40-7　低通滤波器的频率响应

```
>> butterworth low filter
>>
n =

   5

Wn =
   0.0810
```

【例 40-8】 设计一带通滤波器，采样率为 1000 Hz，通带为 60～200 Hz，通带波纹 R_p <=3dB，截止带宽 50 Hz，且在通带两边衰减 40dB。

程序如下：

```
%program p208
Wp = [60 200]/500; Ws = [50 250]/500;
Rp = 3; Rs = 40;
[n,Wn] = buttord(Wp,Ws,Rp,Rs)

[b,a] = butter(n,Wn);
freqz(b,a,128,1000)
title('n=16 Butterworth Bandpass Filter')
```

运行结果如下。滤波特性见图 40-8。

```
>>
n =

  16
```

```
Wn =

    0.1198    0.4005
```

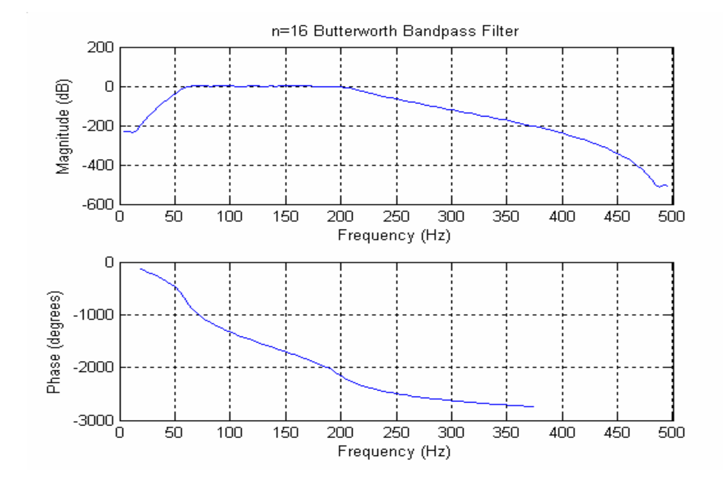


图 40-8　带通滤波器的频率响应

第 41 章　数字滤波器设计

数字滤波器设计工具箱综合了高级的设计、仿真和分析离散时间滤波器技术。通过增加高级的滤波器设计方法和仿真、分析定点和浮点离散时间滤波器技术，从而扩展了信号处理工具箱的能力。

和模拟滤波器不同的是，数字滤波器输入为数字信号，通过数字器件或数值计算方法，对输入信号进行处理，到达保留有用信号去除无用信号的目的。数字滤波器分为 FIR 和 IIR 两种。IIR 和 FIR 滤波器相比，其优点是在满足相同指标下，IIR 滤波器的阶数明显小于 FIR；但是 IIR 滤波器是非线性相位的。下面将分别分析这两种滤波器的设计方法。

41.1　IIR 滤波器设计方法

IIR 滤波器是一种数字滤波器，系统函数为

$$H(z)=\frac{Y(z)}{X(z)}=\sum_{n=0}^{\infty}h(n)z^{-n}=\frac{\sum_{r=0}^{M}b_r z^{-r}}{1+\sum_{k=1}^{N}a_k z^{-k}}$$

IIR 滤波器设计就是根据性能指标要求，设计滤波器的分子和分母多项式。设计方法通常为经典设计法（模拟变换法法）和满足特殊性能的直接设计方法。

41.2　IIR 滤波器经典设计

信号处理工具箱提供的 IIR 经典设计方法是基于经典的低通模拟滤波器到具有相同性能指标的数字滤波器变换。基本原理就是先根据滤波器的技术指标设计出相应的模拟滤波器，然后再将设计好的模拟滤波器变换为满足指标的数字滤波器。下面将介绍怎样设计 IIR 滤波器并归纳详细规范。

41.2.1　IIR 滤波器完全设计函数

用下面的设计函数，你可容易地产生任何阶数的低通、高通、带通和带阻滤波器。见表 41-1。

表 41-1　滤波器设计函数

滤波器类型	设 计 函 数
Bessel (用于模拟滤波器)	[b, a] = besself(n, Wn, options) [z, p, k] = besself(n, Wn, options) [A, B, C, D] = besself(n, Wn, options)
Butterworth	[b, a] = butter(n, Wn, options) [z, p, k] = butter(n, Wn, options) [A, B, C, D] = butter(n, Wn, options)

滤波器类型	设 计 函 数
Chebyshev Type I	[b, a] = cheby1(n, Rp, Wn, options) [z, p, k] = cheby1(n, Rp, Wn, options) [A, B, C, D] = cheby1(n, Rp, Wn, options)
Chebyshev Type II	[b, a] = cheby2(n, Rs, Wn,options) [z, p, k] = cheby2(n, Rs, Wn, options) [A, B, C, D] = cheby2(n, Rs, Wn, options)
椭圆滤波器	[b, a] = ellip(n, Rp, Rs, Wn, options) [z, p, k] = ellip(n, Rp, Rs, Wn, options) [A, B, C, D] =ellip(n, Rp, Rs, Wn, options)

其中，n 为滤波器阶数，Wn 为滤波器的归一化截止频率（Nyquist 频率为 1Hz)；函数默认为低通滤波器。b，a 分别为滤波器传递函数分子和分母系数向量；z，p，k 分别为滤波器的零点、极点和增益。options 为滤波器类型参数：high 为高通滤波器，stop 为带阻滤波器，截止频率 Wn=[W1，W2]。

现分别说明如下。

1．butter 函数

butter 函数用于设计 butterworth 数字滤波器。其调用格式为：

- [b, a] = butter(n, Wn)
- [b, a] = butter(n, Wn, 'ftype')
- [b, a] = butter(n, Wn, 's')
- [b, a] = butter(n, Wn, 'ftype', 's')
- [z, p, k] = butter(...)
- [A, B, C, D] = butter(...)

其中，n 为滤波器阶数，Wn 为滤波器的截止频率，0~1；

'ftype' 为滤波器类型参数：'high' 为高通滤波器，截止频率 Wn；'stop' 为带阻滤波器，截止频率 Wn=[ω_1, ω_2]；默认为低通和带通滤波器。

b，a 分别为滤波器传递函数分子和分母系数向量；z，p，k 分别为滤波器的零、极点和增益。

函数 butter 用于设计数字滤波器时，采用双线性变换法和频率的预畸变处理。将模拟滤波器离散化转换为数字滤波器，同时保证模拟滤波器和数字滤波器在 Wn 或 W_1, W_2 处有相同的幅频响应。设计时要注意模拟和数字频率的转换。

【例 41-1】 设采样率为 1000Hz，设计一个 9 阶高通 Butterworth 滤波器，截止频率为 300 Hz（对应于归一化频率值为 0.6)。

程序如下：

```
%program p301
[b,a] = butter(9,300/500,'high');
freqz(b,a,128,1000); %The filter's frequency response
axis([0 500 -400 100]);%change the axis value to display
```

程序运行结果如图 41-1 所示。

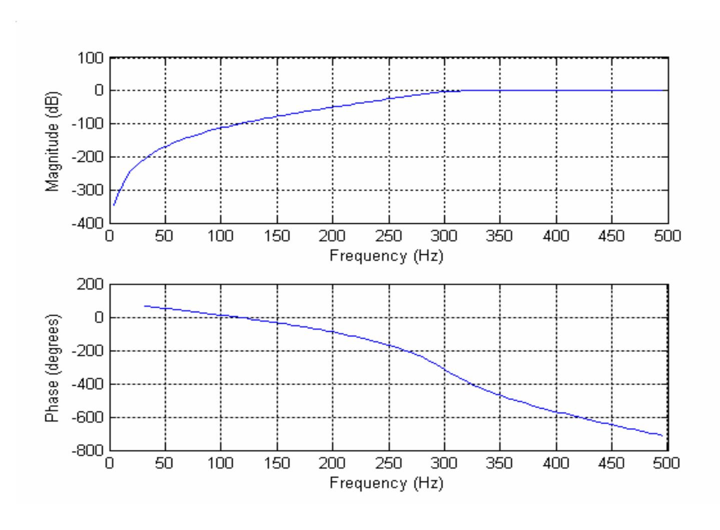


图 41-1　Butterworth 高通滤波器幅频和相频特性

【例 41-2】　设计一个 10 阶的带通 Butterworth 数字滤波器，通带为 100 到 200 Hz，绘出其脉冲响应或单位抽样响应。

程序如下：

```
%program p302
n = 5;
Wn = [100 200]/500;
[b,a] = butter(n,Wn);
[y,t] = impz(b,a,101);
stem(t,y)
```

程序运行结果如图 41-2 所示。

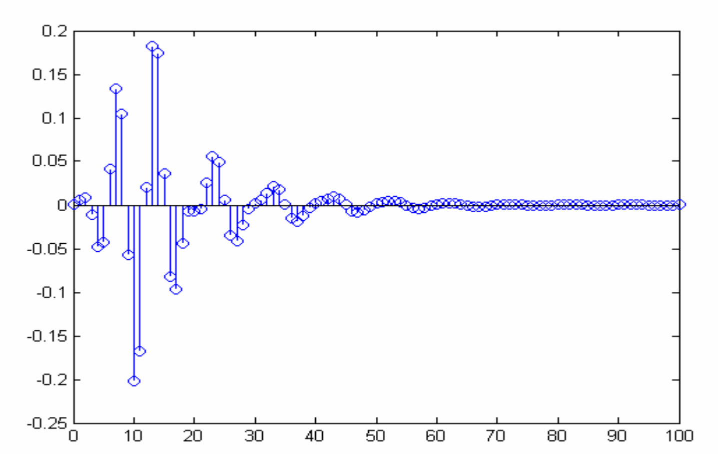


图 41-2　Butterworth 带通滤波器脉冲响应

2. cheby1 函数

cheby1 函数用于设计切比雪夫数字滤波器中的 chebyshev I 型，其调用格式为：

- [b, a] = cheby1(n, Rp, Wn)

- [b, a] = cheby1(n, Rp, Wn, 'ftype')
- [b, a] = cheby1(n, Rp, Wn, 's')
- [b, a] = cheby1(n, Rp, Wn, 'ftype', 's')
- [z, p, k] = cheby1(...)
- [A, B, C, D] = cheby1(...)

格式中，Rp 为通带波纹（dB），Wn 为截止频率，在 0 ~1 之间，在该频率处滤波器的幅值响应为-Rp，其余参数同函数 butter。

【例 41-3】 设采样率为 1000 Hz，设计一个 9 阶低通 Chebyshev I 型数字滤波器，通带波纹 0.5 dB，截止频率为 300 Hz（对应于归一化频率值为 0.6）。

程序如下：

```
%program p303
[b,a] = cheby1(9,0.5,300/500);
freqz(b,a,512,1000);%The frequency response of the filter
axis([0 500 -300 100]);
```

程序运行结果如图 41-3 所示。

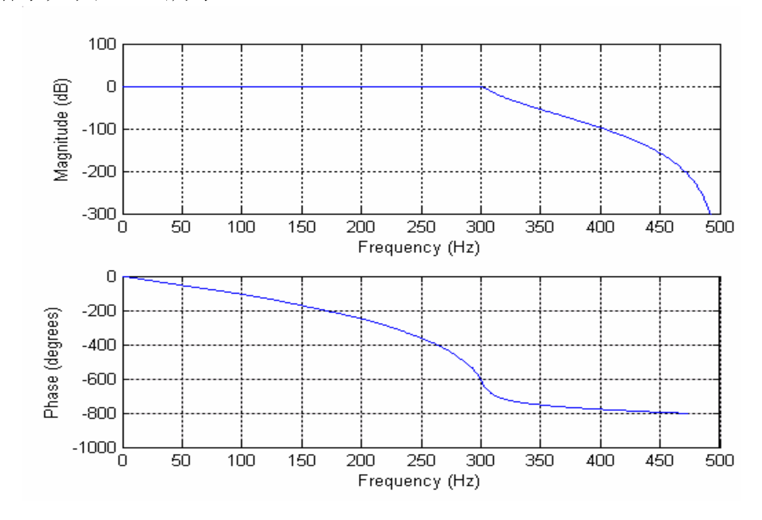


图 41-3　Chebyshev I 型低通滤波器的幅频和相频特性

3．cheby2 函数

cheby2 函数用于设计 chebyshev II 型数字滤波器，其调用格式为：

- [b, a] = cheby2(n, Rs, Wn)
- [b, a] = cheby2(n, Rs, Wn, 'ftype')
- [b, a] = cheby2(n, Rs, Wn, 's')
- [b, a] = cheby2(n, Rs, Wn, 'ftype', 's')
- [z, p, k] = cheby2(...)
- [A, B, C, D] = cheby2(...)

格式中，Rs 为阻带衰减（dB），Wn 为截止频率，在该频率处滤波器的幅值响应为-Rp，其余参数同函数 butter。

【例 41-4】 采样率为 1000 Hz，设计一个 9 阶低通 Chebyshev II 型数字滤波器，阻带衰

减 20 dB，通带波纹 0.5 dB，截止频率为 300 Hz（对应于归一化频率值为 0.6）。

程序如下：

```
%program p304
[b,a] = cheby2(9,20,300/500);
freqz(b,a,512,1000);%The frequency response of the filter
axis([0 500 -80 20]);
```

程序运行结果如图 41-4 所示。

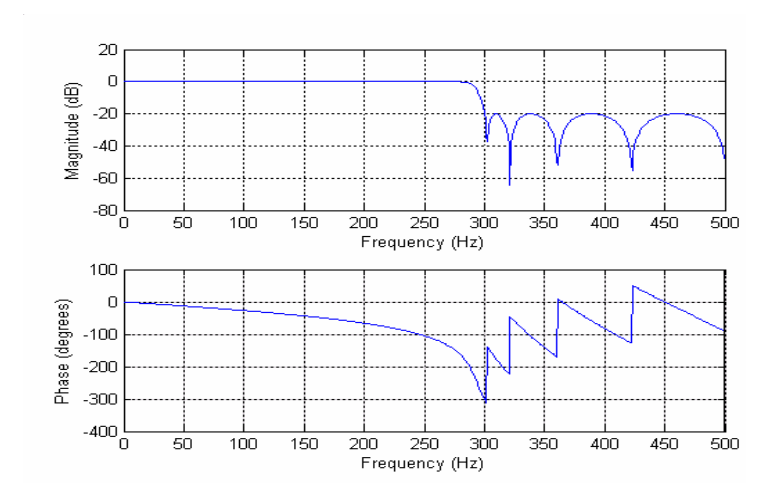


图 41-4　Chebyshev II 型低通滤波器的幅频和相频特性

4．ellip 函数

ellip 函数用于设计椭圆数字滤波器，其调用格式为：

- [b, a] = ellip(n, Rp, Rs, Wn)
- [b, a] = ellip(n, Rp, Rs, Wn, 'ftype')
- [b, a] = ellip(n, Rp, Rs, Wn, 's')
- [b, a] = ellip(n, Rp, Rs, Wn, 'ftype', 's')
- [z, p, k] = ellip(...)
- [A, B, C, D] = ellip(...)

格式中，Rp 为通带波纹（dB），Rs 为阻带衰减（dB），Wn 为截止频率，其余参数同函数 butter。

【例 41-5】　设采样率为 1000 Hz，设计一个 6 阶低通椭圆数字滤波器，阻带衰减 50 dB，通带波纹 3 dB，截止频率为 300 Hz（对应于归一化频率值为 0.6）。

程序如下：

```
%PROGRAM  P305
[b,a] = ellip(6,3,50,300/500);
freqz(b,a,512,1000);%The frequency response of the filter
axis([0 500 -80 20]);
title('n=6 Lowpass Elliptic Filter')
```

程序运行结果如图 41-5 所示。

41.2.2　模拟滤波器变换法

利用上一章模拟滤波器设计方法，设计满足性能的滤波器，离散化为数字滤波器，即将模拟滤波器的系统函数 $H(s)$映射为数字滤波器系统函数 $H(z)$。便完成数字滤波器设计。

实现系统传递函数 s 域至 z 域有两种方法：脉冲响应不变法和双线性映射法。

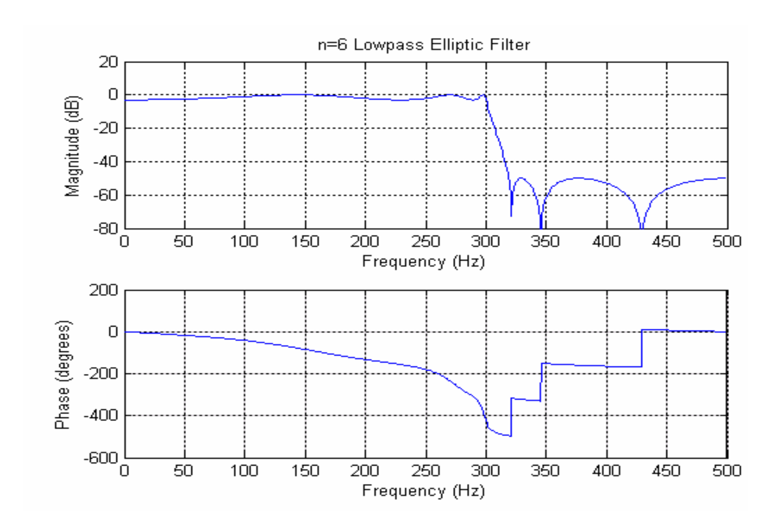


图 41-5　椭圆低通滤波器的幅频和相频特性

1．脉冲响应不变法

$$h(\mathrm{n}) = h_a(t)\big|_{t=nT} = h_a(nT) \tag{41-1}$$

式中 T 为采样周期。因此，数字滤波器的系统函数 H(z)可由下式求得：

$$H(z) = z[h(n)] = z[h_a(nT)] \tag{41-2}$$

如果已经获得了满足性能要求的模拟滤波器的系统函数 $H_a(s)$，求与之对应的数字滤波器的系统函数 $H(z)$ 的方法是：

① 求模拟滤波器的单位冲激响应。

$$h_a(t) = l^{-1}[H_a(s)]$$

② 由式（41-1）求得模拟滤波器的采样值，即数字滤波器脉冲响应系列 $h(n)$。

③ 由式（41-2）求得数字滤波器的系统函数 $H(z)$。

MATLAB 信号处理工具箱提供了实现函数 impinvar，其调用格式为：

- [bz, az] = impinvar(b, a, fs)
- [bz, az] = impinvar(b, a)
- [bz, az] = impinvar(b, a, fs, tol)

其中 b，a 为模拟滤波器的分之和分母多项式系数；fs 为采样频率（Hz），默认为 1；bz，az 分别为数字滤波器分子和分母多项式系数向量。用 tol 容限来限定是否极点重复。

【例 41-6】　用脉冲响应不变法将模拟低通滤波器变换为数字滤波器，采样频率 10 Hz。

程序如下：

```
%PROGRAM  P306
%impulse analog to digital filter conversion
[b,a] = butter(4,0.3,'s');
[bz,az] = impinvar(b,a,10)
```

程序运行结果为：

```
>>
bz =

   1.0e-006 *

  -0.0000    0.1324    0.5192    0.1273         0

az =

   1.0000   -3.9216    5.7679   -3.7709    0.9246
```

2．双线性变换法

MATLAB 为双线性变换法提供了 bilinear 函数。调用格式为：

- [zd, pd, kd] = bilinear(z, p, k, fs)
- [zd, pd, kd] = bilinear(z, p, k, fs, fp)
- [numd, dend] = bilinear(num, den, fs)
- [numd, dend] = bilinear(num, den, fs, fp)
- [Ad, Bd, Cd, Dd] = bilinear(A, B, C, D, fs)
- [Ad, Bd, Cd, Dd] = bilinear(A, B, C, D, fs, fp)

numd，dend 分别为模拟滤波器传递函数分子和分母多项式系数向量。

z，p 分别为模拟滤波器零、极点列向量；k 为模拟滤波器增益；fs 为采样频率，fp（prewarping）为预畸变频率；zd, pd, kd 为数字滤波器的零、极点和增益。

【例 41-7】 用双线性变换法将模拟低通滤波器变换为数字滤波器，采样频率 10 Hz。

程序如下：

```
%program p307
%bilinear analog to digital filter conversion
[b,a] = butter(4,0.3,'s');
[bz,az] = bilinear(b,a,10)
```

程序运行结果为：

```
>> p307
bz =

   1.0e-006 *
   0.0487    0.1947    0.2921    0.1947    0.0487
az =
   1.0000   -3.9216    5.7679   -3.7709    0.9246
```

综上所述，经典滤波器设计方法如下。

（1）根据滤波器的性能和指标，首先对设计性能指标中的频率，如边界频率进行转换，转换后的频率指标作为模拟滤波器原型设计指标。

（2）利用 MATLAB 工具箱函数 buttord，cheblord，cheb2ord，ellipord 等，估计模拟低通滤波器的最小阶数和边界频率。

（3）设计模拟低通滤波器原型，截止频率为 1，并转换原型滤波器指标到期望的滤波器指标。可用 MATLAB 工具箱函数 buttap，cheblap，cheb2ap，ellipap。

（4）频率转换：利用 MATLAB 工具箱函数 lp2lp，lp2hp，lp2bp，lp2bs 等，由模拟低通滤波器经频率变换获得模拟滤波器。

（5）离散化：利用 MATLAB 工具箱函数 bilinear，impinvar，将模拟滤波器离散化得到数字滤波器。

【例 41-8】 用脉冲响应不变法设计一个 Butterworth 低通数字滤波器，使其特性逼近一个 Butterworth 低通模拟滤波器的性能指标如下：通带截止频率 $\Omega_c = 2\pi \times 2\text{k rad/s}$，阻带边界频率 $\Omega_c = 2\pi \times 3\text{k rad/s}$，通带波纹 Rp 小于 3dB，阻带衰减大于 15dB，采样频率 F_s=10000Hz。

程序如下：

```
%program p308
wp=2000*2*pi;
ws=3000*2*pi;
Rp=3;
Rs=15;
Fs=10000;
Nn=128;
[N,Wn]=buttord(wp,ws,Rp,Rs,'s')%compute order and cutoff freq.
[z,p,k]=buttap(N);%compute the analog filter
[Bp,Ap]=zp2tf(z,p,k);
[b,a]=lp2lp(Bp,Ap,Wn);
[bz,az]=impinvar(b,a,Fs)
freqz(bz,az,Nn,Fs)
```

程序运行结果如下：

```
>> p308
N =
    5
Wn =
    1.3387e+004
bz =
   -0.0000    0.0703    0.2923    0.1253    0.0053
az =
    1.0000   -1.1280    0.9642   -0.4467    0.1166   -0.0131
```

滤波器的幅频相频图如图 41-6 所示。

【例 41-9】 用双线性变换设计上述滤波器。

程序如下：

```
%program p309
wp=2000*2*pi;
ws=3000*2*pi;
Rp=3;
Rs=15;
Nn=128;
```

```
Fs=10000;
[N,Wn]=buttord(wp,ws,Rp,Rs,'s')%compute order and cutoff freq.
[z,p,k]=buttap(N);%compute the analog filter
[Bp,Ap]=zp2tf(z,p,k);
[b,a]=lp2lp(Bp,Ap,Wn);
[bz,az]=bilinear(b,a,Fs)
freqz(bz,az,Nn,Fs)
```

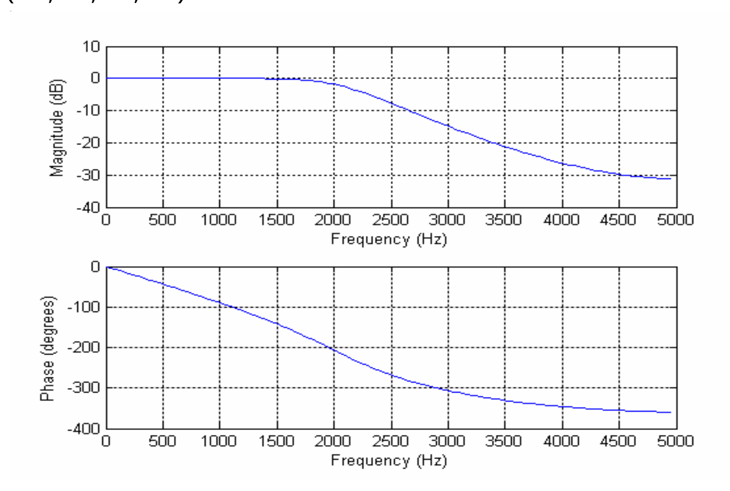


图 41-6　脉冲响应不变法 butterworth 低通滤波器的幅频、相频图

程序运行结果如下：

```
>> p309
N =
    5
Wn =
    1.3387e+004
bz =
    0.0171    0.0854    0.1708    0.1708    0.0854    0.0171
az =
    1.0000   -1.2271    1.1622   -0.5176    0.1450   -0.0159
```

滤波器的幅频相频图如图 41-7 所示：

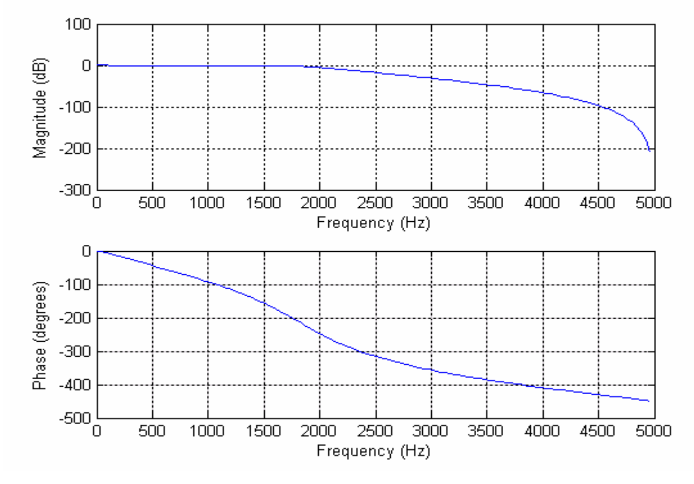


图 41-7　双线性变换 butterworth 低通滤波器的幅频、相频图

【例 41-10】 用脉冲响应不变法设计 Butterworth 低通数字滤波器，要求通带频率为 $0 \leqslant \omega \leqslant 0.2\pi$，通带波纹小于 1dB，阻带在 $0.3\pi \leqslant \omega \leqslant \pi$ 内，幅度衰减大于 15dB，采样周期 $T_s = 0.01\text{s}$。

注意：指标给出的频率均为数字频率，所以设计时应变换为模拟频率指标，以便使用函数。

程序如下：

```
=0.2*pi;
w%program p310
wp s=0.3*pi;
Rp=1;
Rs=15;
Ts=0.01;
Nn=128;
Wp=wp/Ts;%convert digital freq. to analog freq.
Ws=ws/Ts;
[N,Wn]=buttord(Wp,Ws,Rp,Rs,'s')%compute order and cutoff freq.
[z,p,k]=buttap(N);%compute the analog filter
[Bp,Ap]=zp2tf(z,p,k);
[b,a]=lp2lp(Bp,Ap,Wn);
[bz,az]=impinvar(b,a,1/Ts)
freqz(bz,az,Nn,1/Ts)
```

程序运行结果为：

```
>> p310
N =
    6
Wn =
   70.8654

bz =
   -0.0000    0.0007    0.0105    0.0167    0.0042    0.0001
az =
    1.0000   -3.3443    5.0183   -4.2190    2.0725   -0.5600    0.0647
```

滤波器的幅频相频图如图 41-8 所示。

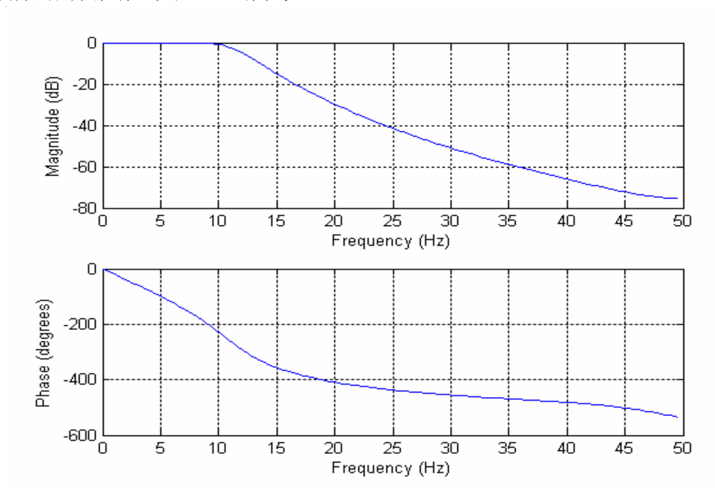


图 41-8 脉冲响应不变法 butterworth 低通滤波器的幅频、相频图

41.3　FIR 滤波器设计方法

FIR 滤波器的系统函数为

$$H(z)=\frac{Y(z)}{X(z)}=\sum_{n=0}^{N-1}h(n)z^{-n}$$

由于 IIR 滤波器设计只能保证其在幅频响应满足性能指标，相位特性无法满足且往往非线性。而 FIR 滤波器的优点是不仅能满足幅频响应要求，而且可以得到线性相位特性，可以用于不失真的信号处理，但是 FIR 滤波器所需要的阶数要高于 IIR 滤波器，所以延迟要大得多。关于 FIR 的相关理论可以参考相关信号处理书籍。

MATLAB 信号处理工具箱提供的 FIR 数字滤波器设计函数用于设计四类具有线性相位的滤波器。下面介绍采用的设计方法和函数。

41.3.1　FIR 窗函数设计

1．基本步骤

基于窗函数的 FIR 数字滤波器设计算法很简单，其主要步骤如下。

（1）由滤波器理想特性 $H_d(\mathrm{e}^{j\omega})$ 进行傅立叶逆变换获得理想滤波器的单位脉冲响应 $h_d(n)$。一般假设理想低通滤波器的截止频率为 ω_c，其幅频特性为

$$H_d(\mathrm{e}^{j\omega})=\begin{cases}1, & 0\leqslant\omega\leqslant\omega_0\\ 0, & \omega_0<\omega<\pi\end{cases}$$

或

$$h_d(n)=\frac{1}{2\pi}\int_{-\omega_c}^{\omega_c}\mathrm{e}^{j\omega n}d\omega=\frac{\sin[\omega_c(n-\alpha)]}{\pi(n-\alpha)}$$

（2）由性能指标确定窗函数 $W(n)$ 和窗口长度 N，由过渡带近似于窗函数主瓣宽度求得窗口长度 N。

（3）求滤波器的单位脉冲响应 $h(n)$

$$h(n)=h_d(n)\cdot W(n)$$

$h(n)$ 即为所设计的 FIR 滤波器系数向量 $b(n)$。

（4）检验滤波器的性能指标。

【例 41-11】　用窗函数设计一线性相位 FIR 低通滤波器，并满足性能指标。通带边界频率 $\omega_p=0.5\pi$，阻带边界频率 $\omega_S=0.66\pi$，阻带衰减不小于 40dB，通带波纹不大于 3dB。

程序如下：

```
%program p311
wp=0.5*pi;
ws=0.66*pi;
width=ws-wp;%width of transition band
N=ceil(8*pi/width);%the length of the filter
if(rem(N,2))==0
   N=N+1;
end
Nw=N;%the length of the filter
wc=(wp+ws)/2;%cutoff freq.of the filter
```

```
n=0:N-1;
alpha=(N-1)/2;
m=n-alpha+0.00001;
hd=sin(wc*m)./(pi*m);
win=hanning(Nw);
h=hd.*win';
b=h;
freqz(b,1,512);
```

程序运行结果如图 41-9 所示。

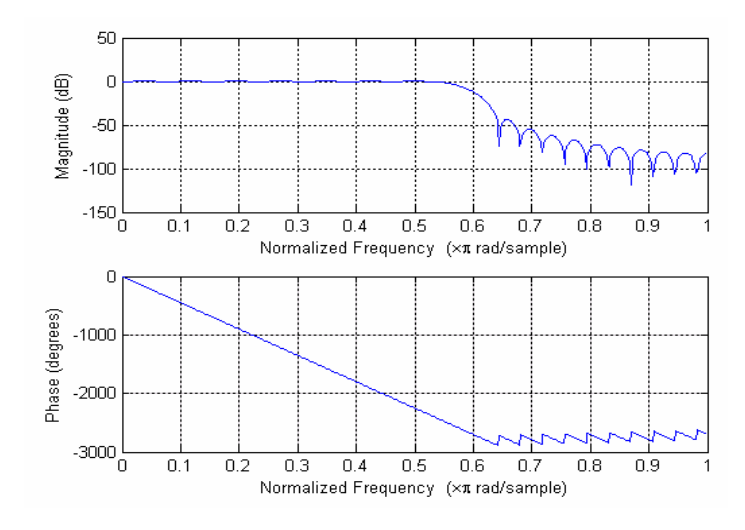


图 41-9 线性相位 FIR 低通滤波器幅频、相频图

2. 工具箱函数

MATLAB 信号处理工具箱还提供了基于窗函数的工具函数 fir1 和 fir2。

（1）fir1 函数。其调用格式为：

- b=fir1(n, ω_n)
- b=fir1(n, ω_n , 'ftype')
- b=fir1(n, ω_n , window)
- b=fir1(n, ω_n , 'ftype'，window)

其中，n 为 FIR 滤波器的阶数，对于高通、带阻滤波器，n 取偶数；ω_n 为滤波器的截止频率，0~1；对于带通、带阻滤波器，$\omega_n=[\omega_1,\omega_2]$，且 $\omega_1<\omega_2$；对于多带滤波器 $\omega_n=[\omega_1,\omega_2,\omega_3,\omega_4]$，频率分段为 $0<\omega<\omega_1$，$\omega_1<\omega<\omega_2$，$\omega_2<\omega<\omega_3$，…

'ftype'为滤波器类型：默认为低通滤波器或带通滤波器；

'high'为高通滤波器；

'stop'为阻带滤波器；

'DC—1'使多带的第一频带为通带；

'DC—0'使多带的第一频带为阻带；

window 为窗函数，列向量，其长度为 n+1。

【例 41-12】 用 FIR1 设计一个 48 阶的 FIR 带通滤波器，通带 $0.35\leqslant\omega\leqslant0.65$。

编程如下：

```
%program p312
b = fir1(48,[0.35 0.65]);
freqz(b,1,512)
```

程序运行结果如图 41-10 所示。

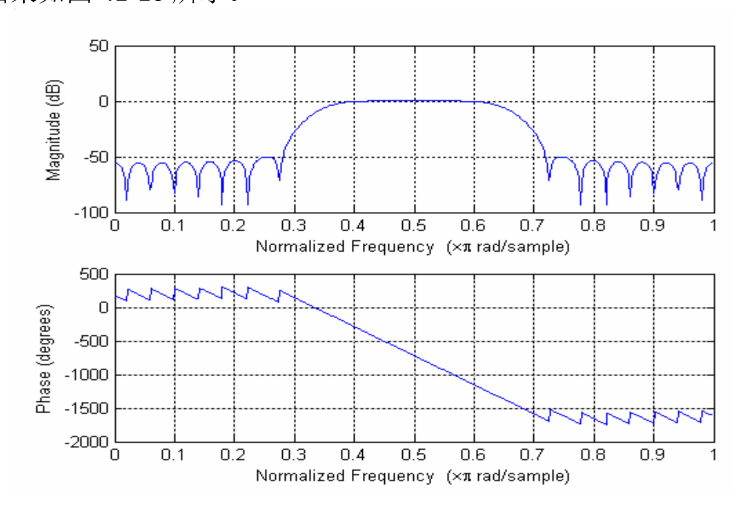


图 41-10　FIR 带通滤波器幅频、相频图

【例 41-13】　设计一个 34 阶的高通滤波器在 fs/4 处衰减，截止频率 0.48，用 30 dB 波纹的 Chebyshev 窗函数。

程序如下：

```
%program p313
load chirp     % Load y and fs.
b = fir1(34,0.48,'high',chebwin(35,30));
freqz(b,1,512)
```

程序运行结果如图 41-11 所示。

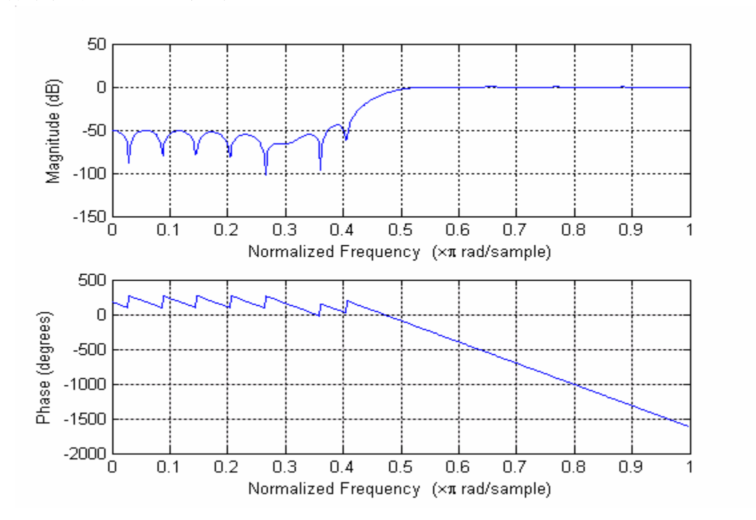


图 41-11　FIR 高通滤波器的幅频、相频图

（2）fir2 函数，其调用格式为：

- b = fir2(n, f, m)

- b = fir2(n, f, m, window)
- b = fir2(n, f, m, npt)
- b = fir2(n, f, m, npt, window)
- b = fir2(n, f, m, npt, lap)
- b = fir2(n, f, m, npt, lap, window)

其中，n 为滤波器阶数；f 和 m 分别为滤波器期望幅频响应的频率向量和幅值向量，取值在 0~1 间，m 和 f 长度相同；window 为窗函数，列向量，长度必须是（n+1）；默认时自动取 hamming；npt 为对频率响应进行内插点数，默认时为 512；lap 定义一个区域尺寸，默认为 25；b 为 FIR 滤波器系数向量，长度为 n+1。

【例 41-14】 设计一个 30 阶低通滤波器并绘出期望频率响应和实际频率响应。

程序如下：

```
%program p314
f = [0 0.6 0.6 1];
m = [1 1 0 0];
b = fir2(30,f,m);
[h,w] = freqz(b,1,128);
plot(f,m,w/pi,abs(h))
legend('Ideal','fir2 Designed')
title('Comparison of Frequency Response Magnitudes')
```

程序运行结果如图 41-12 所示。

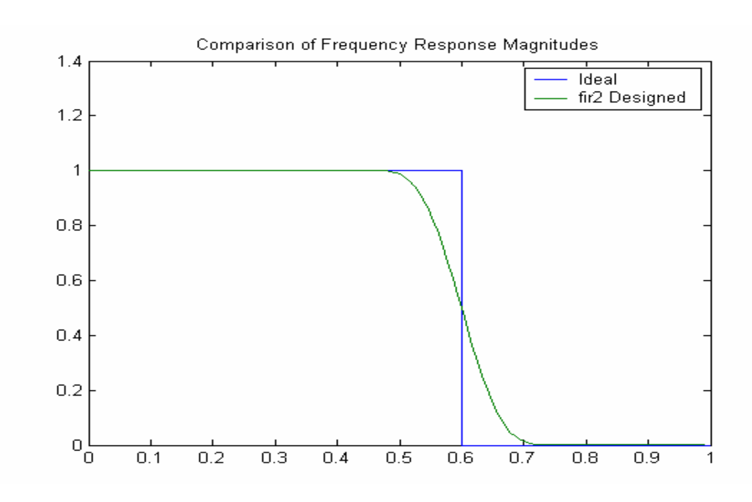


图 41-12 FIR 低通滤波器幅频响应

41.3.2 最优 FIR 滤波器设计

1．有关函数介绍

MATLAB 信号处理工具箱函数提供了比基于窗函数法 FIR 滤波器设计工具箱函数 fir1 更为通用的函数：firls，remez。

它们采用不同的优化方法设计最优的标准的和多频带的数字 FIR 滤波器。

函数 firls 是 fir1 和 fir2 的扩展，基本准则是利用最小二乘法使期望的频率响应和实际的

频率响应间的误差最小。

函数 remez 实现 Parks-McCellan 算法，这种算法利用 remez 交换算法和 Chebyshev 近似理论来设计滤波器，使实际频率响应拟合期望频率响应最优。

函数调用格式如下：

- b=firls(n, f, a)
- b=remez(n, f, a)

其中，n 为滤波器阶数；f 为滤波器期望频率特性的频率向量标准化频率，0~1，递增向量，允许定义重复频点；a 为滤波器期望频率特性的幅值向量，向量 a 和 f 必须同长度且为偶数；b 为函数返回的滤波器系数，长为 n+1，且具有偶对称关系

$$b(k)=b(n+2-k)$$

2．应用实例

下例分别用两种方法设计 FIR 滤波器。

【例 41-15】 设计一 24 阶的反对称分段线性带通滤波器，并绘出期望的实际频率响应。其中理想幅频响应对为：

```
F = [0  0.3  0.4  0.6  0.7  0.9];
A = [0  1    0    0    0.5  0.5];
```

用 firls 函数设计程序如下：

```
%program p315
F = [0  0.3  0.4  0.6  0.7  0.9];
A = [0  1    0    0    0.5  0.5];
b = firls(24,F,A,'hilbert');
for i=1:2:6,
   plot([F(i) F(i+1)],[A(i) A(i+1)],'--'), hold on
end
[H,f] = freqz(b,1,512,2);
plot(f,abs(H)), grid on, hold off
legend('Ideal','firls Design')
```

程序运行结果如图 41-13 所示。

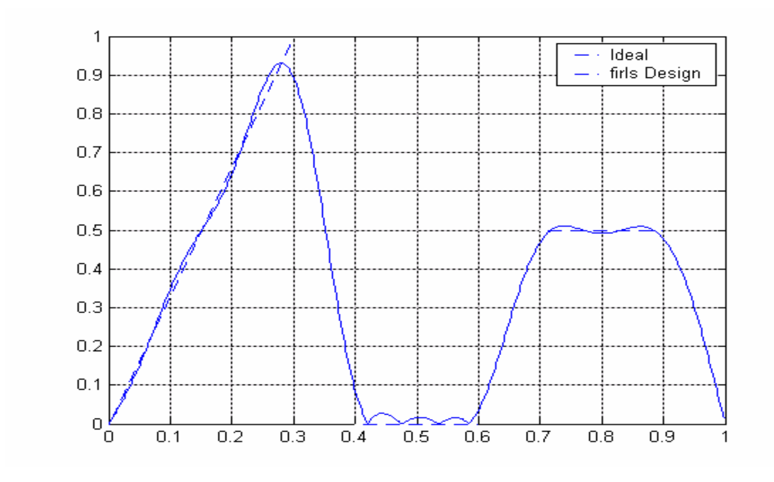


图 41-13　FIR 反对称分段线性带通滤波器频率响应

用 remez 函数设计程序如下：

```
%program p316
F = [0 0.3  0.4 0.6  0.7 0.9];
A = [0  1   0  0  0.5 0.5];
b = remez(24,F,A);
for i=1:2:6,
   plot([F(i) F(i+1)],[A(i) A(i+1)],'--'), hold on
end
[H,f] = freqz(b,1,512,2);
plot(f,abs(H)), grid on, hold off
axis([0 1 0 1]);
legend('Ideal','firls Design')
```

程序运行结果如图 41-14 所示。

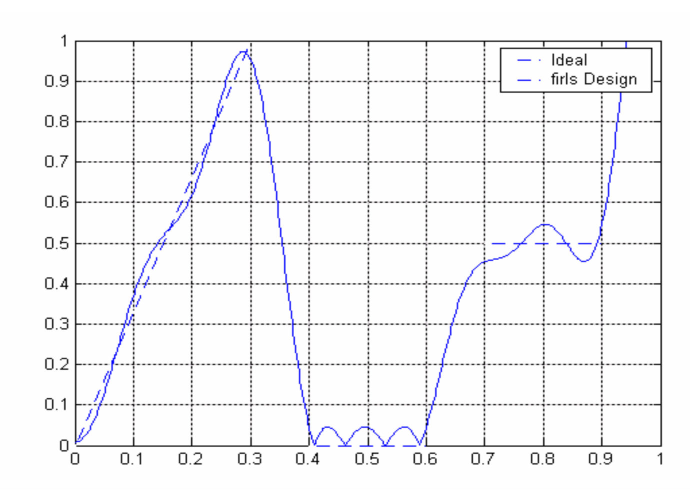


图 41-14　FIR 反对称分段线性带通滤波器频率响应

第 42 章　滤波器分析

由前面的介绍，我们根据指标设计了数字滤波器，得到了滤波器的传递函数 $H(z)$。如何把传递函数 $H(z)$变成具体的数字系统，是滤波器的实现问题。下面将介绍用 MATLAB 函数来分析滤波器的时域和频域特性。

MATLAB 信号处理工具箱提供了许多工具函数来分析滤波器特性，包括时域分析，对任意输入响应，脉冲响应；频域分析，幅值响应，相位响应，零极点位置；其他特性，群延迟等。滤波器时域和频域分析是设计各类滤波器、评价滤波器性能和滤波器应用的基础。下面将介绍这些工具函数。

42.1　时间响应

MATLAB 信号处理工具箱常用的滤波器时间响应分析工具函数有 filter，fftfilt，impz 等。

1．filter 函数

该函数用于实现 IIR 和 FIR 滤波器对数据滤波，常用来计算滤波器对输入的响应。调用格式为：

- $y = filter(b,a,x)$
- $[y,z_f] = filter(b,a,x)$
- $[y,z_f] = filter(b,a,x,z_i)$

其中，b，a 分别是滤波器传递函数 $H(z)$ 的分子和分母多项式系数向量，x 为滤波器输入，向量或者矩阵；若 x 为方阵、每组数据应为列向量；z_i 为状态向量初值，z_f 为状态向量终止值；y 为滤波器输出。

【例 42-1】　产生有 3 个正弦分量（5Hz，15Hz 和 30Hz）的信号。然后，设计一滤波器来去除 5Hz 和 30Hz 的正弦信号，保留 15Hz 信号。

程序如下：

① 产生含有 3 个正弦分量的信号。

```
%program p401
Fs = 100;
t = (1:100)/Fs;
s1 = sin(2*pi*t*5); s2=sin(2*pi*t*15); s3=sin(2*pi*t*30);
s = s1+s2+s3;
plot(t,s);
xlabel('Time (seconds)'); ylabel('Time waveform');
```

程序运行结果如图 42-1 所示。

② 产生一个 8 阶的 IIR 带通滤波器，通带 10 Hz 到 20 Hz，其频率响应如下：

```
%program p402
[b,a] = ellip(4,0.1,40,[10 20]*2/Fs);
```

```
[H,w] = freqz(b,a,512);
plot(w*Fs/(2*pi),abs(H));
xlabel('Frequency (Hz)'); ylabel('Mag. of frequency response');
grid;
```

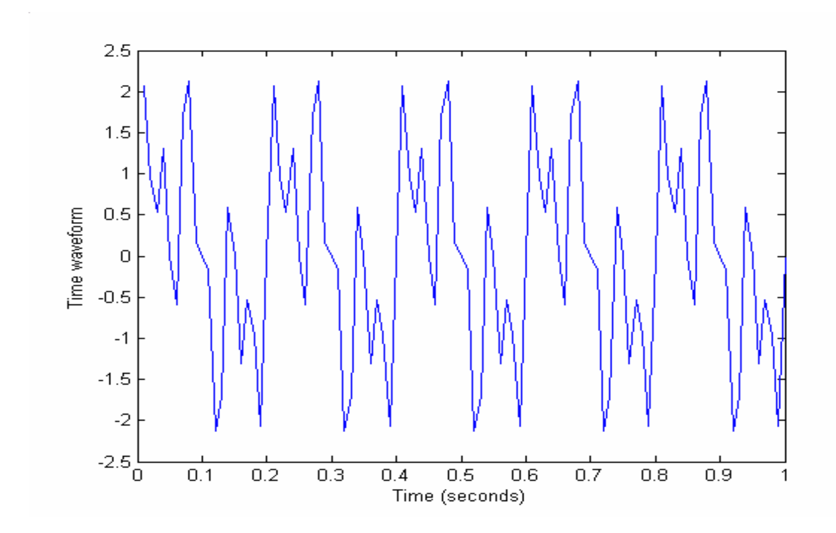


图 42-1　含有 3 个正弦分量的信号

程序运行结果如图 42-2 所示。

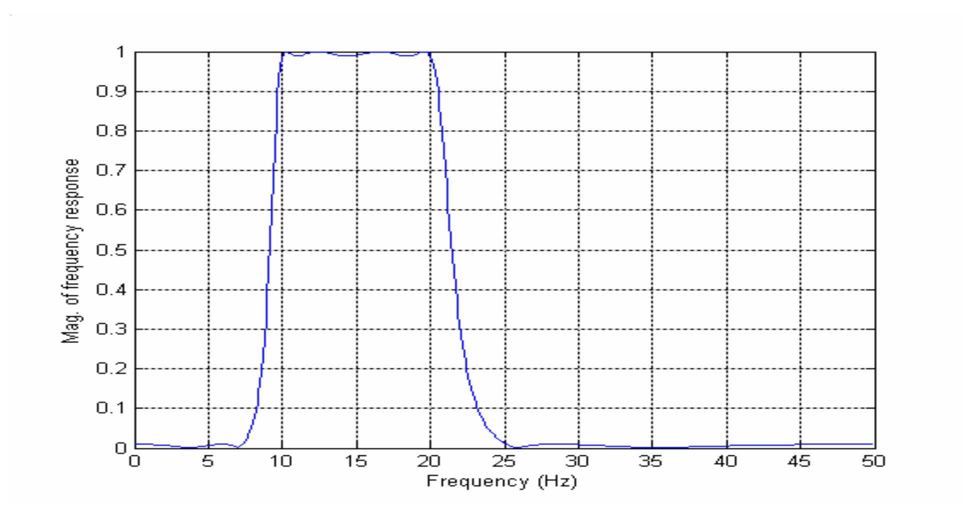


图 42-2　IIR 带通滤波器幅频响应

③ 对信号进行滤波。

```
%program p403
sf = filter(b,a,s);
plot(t,sf);
xlabel('Time (seconds)');
ylabel('Time waveform');
axis([0 1 -1 1]);
```

程序运行结果如图 42-3 所示。

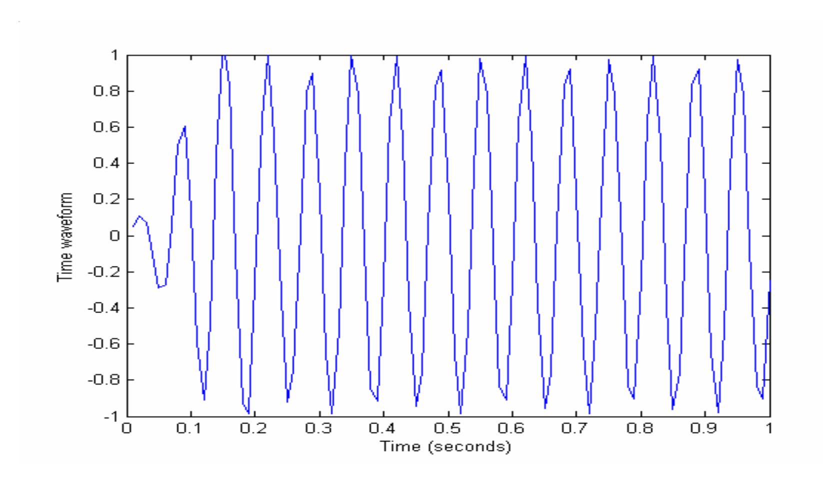


图 42-3　滤波后的信号波形

④ 绘出信号滤波前、后的幅频图

```
%program p404
S = fft(s,512);
SF = fft(sf,512);
w = (0:255)/256*(Fs/2);
plot(w,abs([S(1:256)' SF(1:256)']));
xlabel('Frequency (Hz)'); ylabel('Mag. of Fourier transform');
grid; legend({'before','after'})
```

程序运行结果如图 42-4 所示。

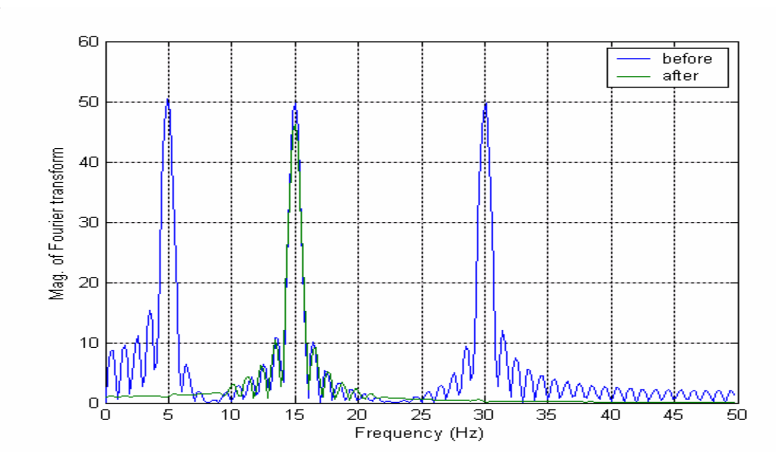


图 42-4　滤波前、后的频幅图

2．fftfilt 函数

该函数利用效率高的基于 FFT 重叠相加算法实现对数据滤波，该函数只适用于 FIR 滤波器，其调用格式为：

- y = fftfilt(b, x)
- y = fftfilt(b, x, n)

其中，b 为 FIR 滤波器系数向量；x 为输入数据；n 为 FFT 长度，默认时，函数选择最佳

的 FFT 长度；y 为滤波器输出。该函数在频域内实现 FIR 滤波器，采用下面的 FFT 过程：

```
n=length(x);
y=ifft(fft(x).*fft(b,n)./fft(a,n))
```

y=fftfilt(b,x)等价于 y=filter(b,1,x)。

3．Impz 函数

函数 impz 用于产生数字滤波器的脉冲响应，其调用格式为：

- [h,t] = impz(b,a)
- [h,t] = impz(b,a,n)
- [h,t] = impz(b,a,n,F_s)
- impz(b,a)

其中，b，a 分别为滤波器分子和分母多项式系数向量；n 为采样点数；F_s 为采样周期，默认为 1；h 为滤波器单位脉冲响应向量；t 为与 h 对应的时间向量。

【例 42-2】 绘出 14 阶低通椭圆滤波器的前 50 采样点的单位脉冲响应，截止频率为 0.4Nyquist 频率。

程序如下：

```
%program p405
[b,a] = ellip(4,0.5,20,0.4);
impz(b,a,50)
```

程序运行结果如图 42-5 所示。

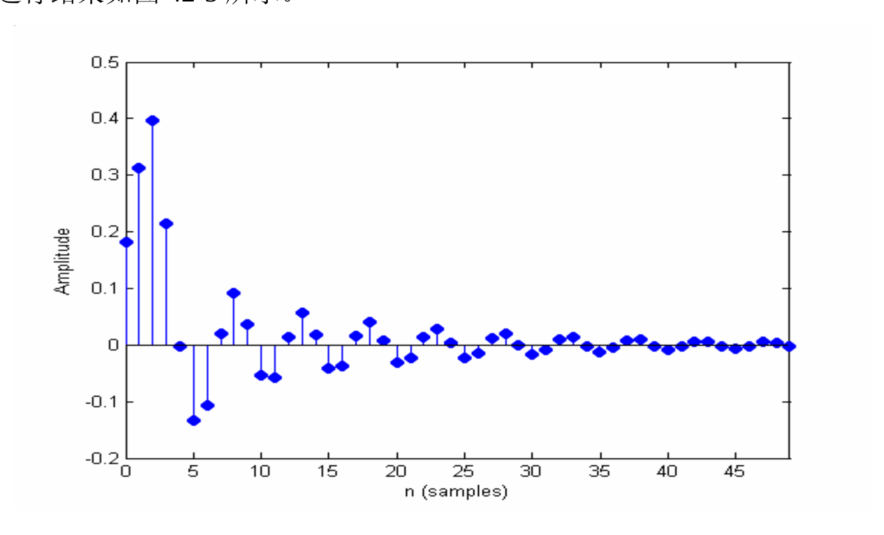


图 42-5 椭圆滤波器前 50 采样点的单位脉冲响应

在 MATLAB 中，[h,t] = impz(b,a,n) 等价于下面的语句：

x = [1 zeros(1,n)]；

h = filter(b,a,x)；

42.2 频率响应

滤波器的频率分析函数主要是 freqs 和 freqz 等。

1．freqs 函数

函数 freqs 用于求模拟滤波器的频率响应，其调用格式为：

- $h = freqs(b,a,\omega)$
- $[h,\omega] = freqs(b,a)$
- $[h,\omega] = freqs(b,a,n)$
- $freqs(b,a)$

其中，b，a 分别为模拟滤波器传递函数分子和分母多项式系数向量；n 为频率点数，默认为 200；h 为频率响应，复数；ω 为频率向量，实数。

【例 42-3】 求出并绘出下面传递函数的幅频响应和相频响应。

$$H(s) = \frac{0.2s^2 + 0.3s + 1}{s^2 + 0.4s + 1}$$

程序如下：

```
%program p406
a = [1 0.4 1];
b = [0.2 0.3 1];
w = logspace(-1,1);
freqs(b,a,w)
```

程序运行结果如图 42-6 所示。

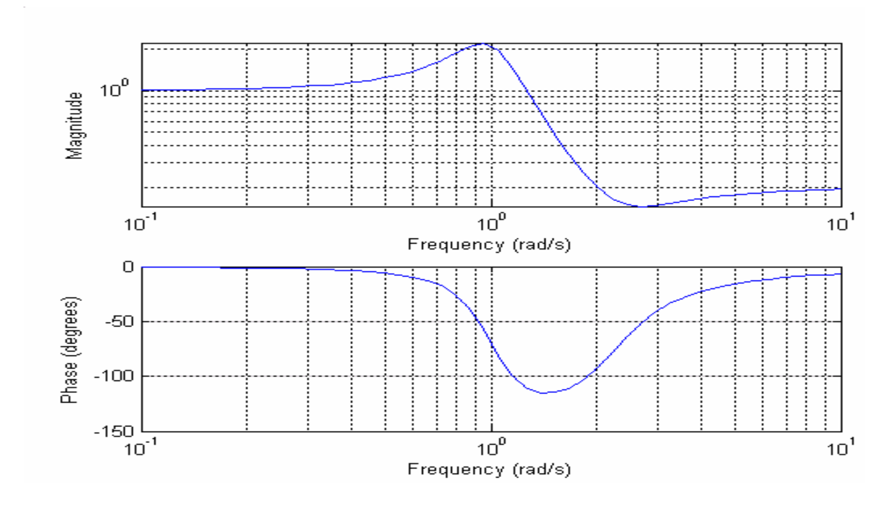


图 42-6　滤波器的幅频响应和相频响应

2．freqz 函数

函数 freqz 用于求数字滤波器的频率响应，其调用格式为：

- $[h,\omega] = freqz(b,a,n)$
- $[h,f] = freqz(b,a,n,F_s)$
- $[h,\omega] = freqz(b,a,n,'whole')$
- $[h,f] = freqz(b,a,n,'whole',F_s)$
- $h = freqz(b,a,\omega)$
- $h = freqz(b,a,f,F_s)$

● freqz(b,a)

其中，函数输入：b，a 分别为数字滤波器 z 的传递函数分子和分母多项式系数向量；n 为复频率响应计算点数，整数，最好为 2 的幂，默认为 512；F_s 为采样频率，f 为给定的频率矢量；'whole' 表示返回的ω值为包含 z 平面整个单位圆频率矢量，即 0~2π；默认时，ω仅包含 z 平面上半单位圆间等间距 N 个点频矢量。

函数返回：h 为复频率响应；ω为点频率向量（弧度），返回ω范围与输入参数 'whole' 有关；f 为 n 点频率向量。

3. unwarp 函数

函数 freqz 输出的频率向量ω在 0～2π 之间，为了获得一个滤波器真正的相频特性图，要对相位ω进行修正。为此需要利用 MATLAB 工具箱函数 unwarp。

函数 unwarp 用于展开函数 freqz 产生的频率ω。调用格式为：

● P=unwarp(ω)

【例 42-4】 绘出下面 FIR 滤波器的幅频响应和相频响应。

程序如下：

```
%program p407
b = fir1(80,0.5,kaiser(81,8));
freqz(b,1);
```

程序运行结果如图 42-7 所示。

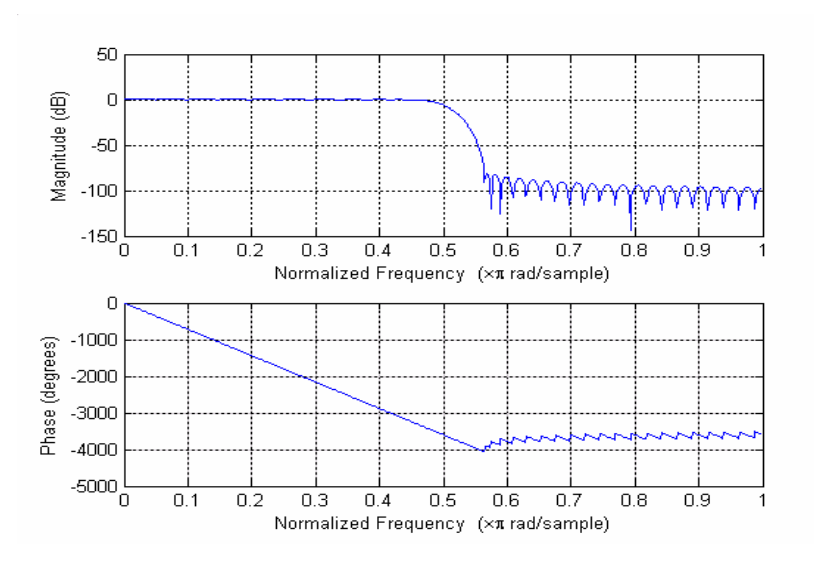


图 42-7 FIR 滤波器的幅频响应和相频响应

42.3 零极点图

1. 有关函数介绍

滤波器零极点位置决定了滤波器稳定性和性能，因此，考察滤波器零极点位置是分析滤波器特性的重要方面。MATLAB 提供的零极点图的工具函数是 zplane 函数。

该函数调用格式为：

- zplane(z, p)
- zplane(b, a)

其中，z，p 分别为系统零、极点向量；b，a 为系统传递函数分子、分母多项式系数向量。

2．应用实例

【例 42-5】 设采样率为 1000 Hz，绘出 5 阶椭圆低通数字滤波器零极点图，其截止频率为 200 Hz，通带波纹 3 dB，阻带衰减 30dB。

程序如下：

```
%program p408
[z,p,k] = ellip(4,3,30,200/500);
zplane(z,p);
title('4th-Order Elliptic Lowpass Digital Filter');
```

程序运行结果如图 42-8。

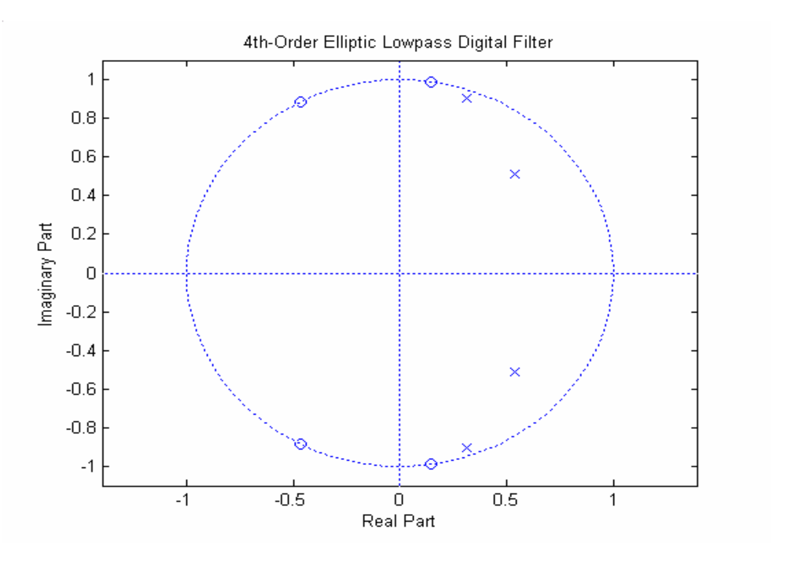


图 42-8　椭圆低通数字滤波器零极点图

42.4　相时延

1．有关函数介绍

phasedelay 函数计算数字滤波器的相时延，其调用格式为：

- [phi, w] = phasedelay(b, a, n)
- [phi, w] = phasedelay(b, a, n, 'whole')
- phi = phasedelay(b, a, w)
- [phi,f] = phasedelay(b, a, n, fs)
- [phi,f] = phasedelay(b, a, n, 'whole', fs)
- phi = phasedelay(b, a, f, fs)
- [phi, w, s] = phasedelay(...)

● [phi, f, s] = phasedelay(...)

其中，phi 为相时延；其他各项意义同函数 freqz。函数输出默认时，绘出延迟图。

2．应用实例

【例 42-6】　绘出椭圆滤波器的相时延。

程序如下：

```
%program p409
[b,a] = ellip(10,.5,20,.4);
phasedelay(b, a, 512, 'whole')
```

运行结果见图 42-9。

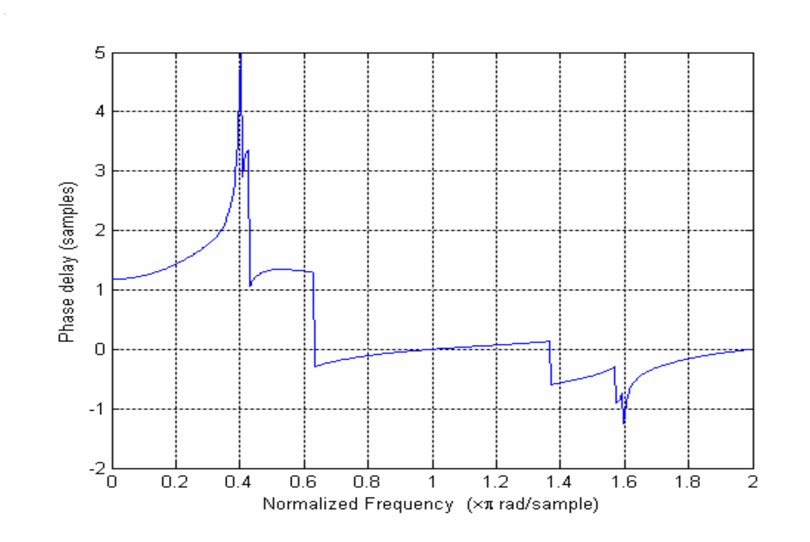


图 42-9　椭圆数字滤波器相时延图

42.5　群延迟

1．有关函数介绍

由信号传输不失真条件知，滤波器相频特性应该是一条经过原点的直线。MATLAB 信号处理工具箱提供了计算群延迟函数 grpdelay，其调用格式为：

- $[gd,\omega]=grpdelay(b,a,n)$
- $[gd,f]=grpdelay(b,a,n,F_s)$
- $[gd,\omega]=grpdelay(b,a,n,'whole')$
- $[gd,f]=grpdelay(b,a,n,'whole',F_s)$
- $gd=grpdelay(b,a,\omega)$
- $gd=grpdelay(b,a,f,F_s)$
- grpdelay

其中，gd 为群延迟；其他各项意义同函数 freqz。函数输出默认时，绘出延迟图。

2．应用实例

【例 42-6】　绘出 6 阶 Butterworth 滤波器 b(z)/a(z)的群延迟图。截止频率 0.2。

程序如下：

```
%program p410
[b,a] = butter(6,0.2);
grpdelay(b,a,128)
```

程序运行结果如图 42-10 所示。

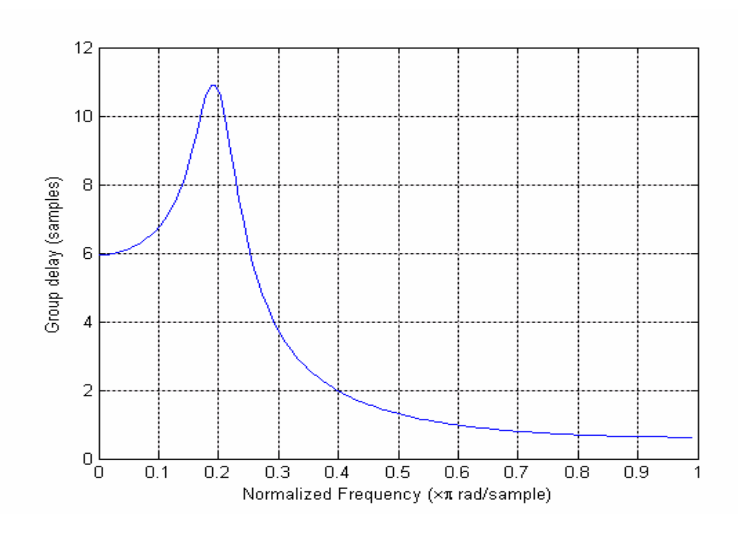


图 42-10　Butterworth 滤波器群延迟图

第 43 章　随机信号的参数模型和功率谱估计

随机信号不能用确定的数学关系式来描述，无法用确定信号的处理方法进行处理。通常用其样本的统计特征来描述。本章将介绍随机过程的相关分析（时域）和功率谱（频域）的基本方法及 MATLAB 实现。

43.1　相关函数的估计

1．相关函数介绍

MATLAB 工具箱提供了计算随机信号相关函数：xcorr。

函数 xcorr 用于计算随机信号的自相关和互相关函数。调用格式如下。

- $\mathrm{c = xcorr(x, y)}$
- $\mathrm{c = xcorr(x, y, 'option')}$
- $\mathrm{c = xcorr(x, y, max\,lags, 'option')}$
- $\mathrm{[c, lags] = xcorr(x, y, max\,lags, 'option')}$

其中，x，y 为两个独立的随机信号系列，长度均为 N 的向量；c 为 x，y 的互相关函数估计。

'option' 为选择项：

① 'biased'，计算有偏互相关估计。

$$c_{xy,biased}(m) = \frac{1}{N} c_{xy}(m)$$

② 'unbiased'，计算无偏互相关估计。

$$c_{xy,unbiased}(m) = \frac{1}{N - |m|} c_{xy}(m)$$

③ 'coeff'，系列归一化，使零延迟的自相关为 1。

④ 'none'，默认情况，函数执行非归一化计算相关。

maxlags 为 x 和 y 之间的最大延迟，函数返回值 c 为 2maxlags+1。若默认，则函数返回值 c 长度为 2N−1；

若函数用于计算 $x(n)$的自相关，则调用格式为：

- c=xcorr(x)
- c=xcorr(x, maglags)

2．应用实例

【例 43-1】　求受白噪声干扰的正弦信号和白噪声信号的自相关并比较。

程序如下：

```
%program p501
clf  %clear the variable
N=1000;
n=0:N-1;
Fs=500;
```

```
t=n/Fs;
Lag=100;
x=sin(2*pi*10*t)+0.6*randn(1,length(t));
[c,lags]=xcorr(x,Lag,'unbiased');
subplot(221);
plot(t,x)
xlabel('t');
ylabel('x(t)')
title('orignal signal x');
grid;
subplot(222);
plot(lags/Fs,c);
xlabel('t');
ylabel('Rx(t)')
title('autocorrelation');
grid;
x1=randn(1,length(x));
[c,lags]=xcorr(x1,Lag,'unbiased');
subplot(223);
plot(t,x1)
xlabel('t');
ylabel('x1(t)')
title( 'white_noise');
grid;
subplot(224);
plot(lags/Fs,c);
xlabel('t');
ylabel('Rx1(t)')
title('autocorrelation');
grid;
```

运行结果如图 43-1 所示。

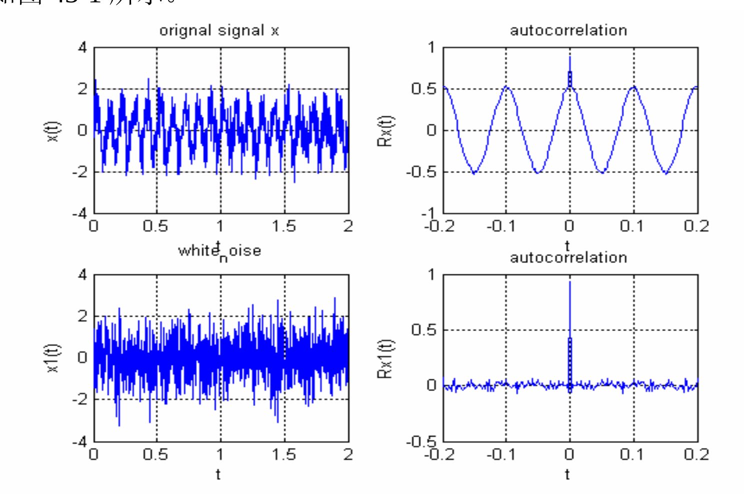


图 43-1　正弦信号和白噪声信号的自相关

由图可见，含有周期分量和干扰噪声的信号的自相关函数在$\tau=0$处具有最大值，且在τ较大时仍具有明显周期性，其频率和周期分量的频率相同。而白噪声信号在$\tau=0$处也具有最大值，但在τ稍大时明显减小至零。自相关函数的此性质可以用来识别随机信号中是否具有周期信号分量。

【例 43-2】 已知两周期信号

$$x(t)=\sin(2\pi ft),\ \ y(t)=0.5\sin(2\pi ft+90°)$$

其中，$f=10\text{Hz}$，求互相关函数 $R_{xy}(\tau)$。

程序如下：

```
%program p502
clf  %clear the variable
N=1000;
n=0:N-1;
Fs=500;
t=n/Fs;
Lag=200;
x=sin(2*pi*10*t);
y=0.5*sin(2*pi*10*t+90*pi/180);
[c,lags]=xcorr(x,y,Lag,'unbiased');
subplot(311);
plot(t,x,'r')
xlabel('t');
ylabel('x(t)');
title('orignal signal');
subplot(312)
plot(t,y,'b')
xlabel('t');
ylabel('y(t)');
title('orignal signal');
grid;
subplot(313);
plot(lags/Fs,c,'r');
xlabel('t');
ylabel('Rxy(t)')
title('correlation');
grid;
```

运行结果如图 43-2 所示。

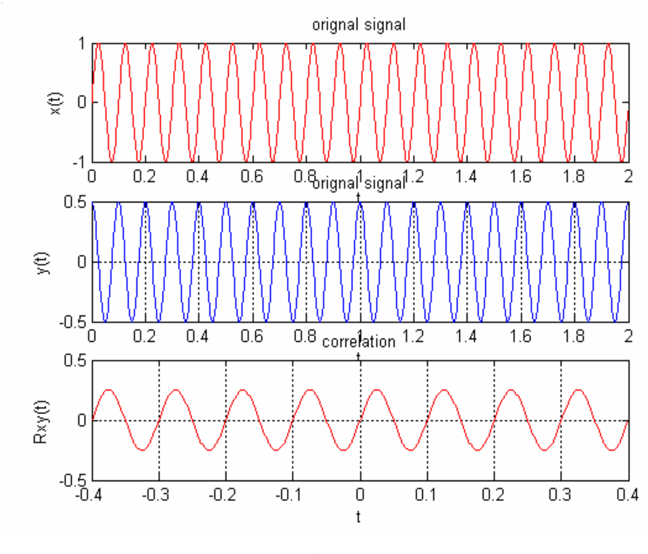


图 43-2 信号的互相关

43.2　经典功率谱估计

1．基本方法

经典功率谱估计如下。

（1）由 $x(n)$ 的 N 个样本估计自相关函数 $\hat{r}_x(m)$ 。

$$\hat{r}_x(m)=\sum_{-\infty}^{\infty}x(n)x^*(n-m)=\sum_{n=1}^{N}x(n)x^*(n-m)\frac{1}{N}$$

（2）估计功率谱 $\hat{p}_x(\omega)$

$$\hat{p}_x(\omega)=\sum_{m=0}^{N}\hat{r}_x(m)\mathrm{e}^{-j\omega m}$$

2．应用实例

【例 43-3】 $x(n)=\exp(j\pi n-j\pi)+\exp(j\omega_0 n-j0.7\pi)]+v(n)$ 为一复正弦加白噪声随机过程,其中 $v(n)$ 为零均值白噪声， $S/N=10\mathrm{dB}$ 。要求：① 产生仿真数据，② 估计自相关函数，③ 估计经典功率谱。

（1）产生仿真数据 N=1000 点，x(1)，x(2)…x(N)， ω =1.4 π ，程序如下：

```
%program p503
var=sqrt(1/exp(1.0));
v=var*randn(1,1000);
n=1:1000;
w0=1.4*pi;
xn=exp(j*pi*n-j*pi)+exp(j*w0*n-j*0.7*pi)+v;
plot(n,xn);
xlabel('n');
title('x(n)=exp(j*pi*n-j*pi)+ exp(jwn-j*0.7*pi)+v(n)');
```

运行结果如图 43-3 所示。

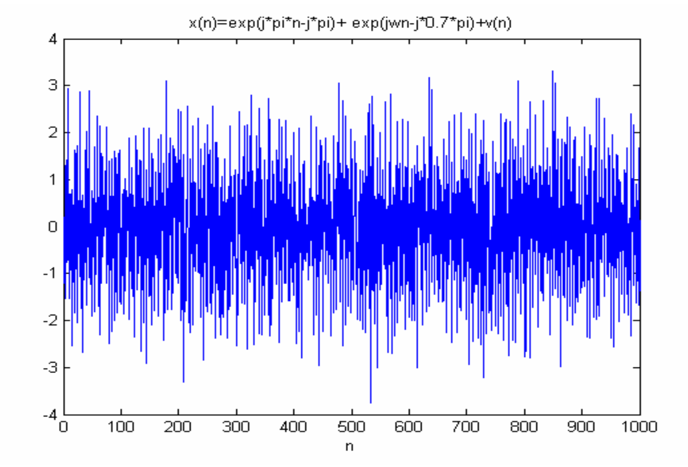


图 43-3　正弦信号加白噪声

（2）自相关函数为 $\hat{r}(m)=\dfrac{1}{N}\sum_{n=m}^{N}x(n)x^*(n-m)$ ， m=0, 1…500。程序如下：

```
%program p504
%corr
```

```
m=-500:500;
[r,lag]=xcorr(xn,500,'biased');
hndl=stem(m,r);
set(hndl,'Marker','o');
set(hndl,'MarkerSize',2);
xlabel('n');
title('自相关 r');
```

运行结果如图 43-4 所示。

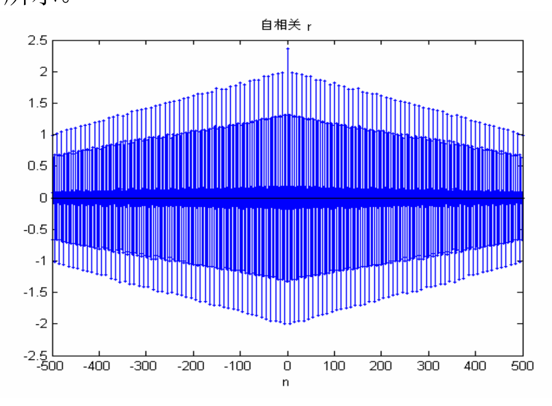


图 43-4　信号自相关

（3）估计经典功率谱（BT）$P_{BT}(\omega)=\sum_{m=-500}^{500}\hat{r}(m)\mathrm{e}^{-j\omega m}$，程序如下：

```
%program p505
%blackman-pbt
k=0:1000;
w=(pi/500)*k;
M=k/500;
X=r*(exp(-j*pi/500)).^(m'*k);
magX=abs(X);
plot(M,10*log10(magX));
xlabel('Angular frequency,ω (π)');
ylabel('Power Spectrum Magnitude (dB)');
title('Blackman-Turkey Spectral Estimate')
```

运行结果如图 43-5 所示。

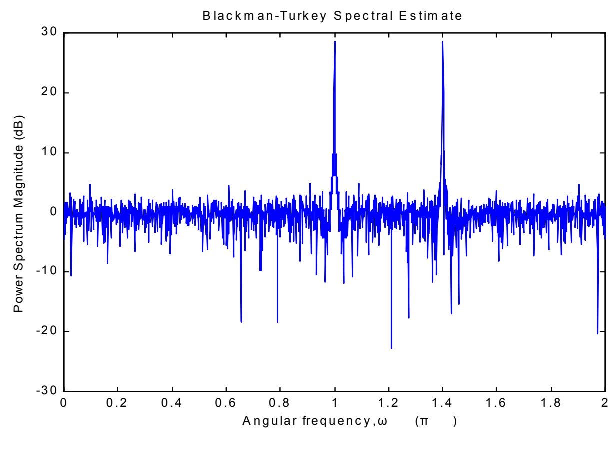


图 43-5　信号的功率谱估计

43.3　AR 模型功率谱估计

1．基本原理

AR 模型的功率谱为

$$P_x(\omega)=\frac{{\sigma_v}^2}{|H(\omega)|^2}=\frac{{\sigma_v}^2}{\left|1+a_1\mathrm{e}^{-j\omega}+a_2\mathrm{e}^{-j2\omega}+\cdots+a_p\mathrm{e}^{-jp\omega}\right|}$$

待估计参数为：$a_1,a_2\cdots a_p,\sigma_v^2$，共 p+1 个，由 AR 模型的正则方程可以解出参数。

2．应用实例

【例 43-4】 设有二阶 AR 模型 $u(n)+a_1u(n-1)+a_2u(n-2)=v(n)$，$v(n)$ 为 0 均值，方差 σ_v^2 的白噪声，$\sigma_u^2=1$，$a_1=-0.195$，$a_2=0.95$。① 产生 N 个二阶 AR 随机过程的样本值，$u(1),u(2),\cdots,u(N),N=1000$，②根据 $u(1),u(2),\cdots,u(N)$ 估计 $\hat{r}(m)$，③计算 $\hat{a}_1,\hat{a}_2,{\sigma_v}^2$，④画出功率谱 $P_{BT}(\omega)=\sum_{m=-255}^{255}r(m)\mathrm{e}^{-j\omega n}$，$P_{AR}(\omega)=\frac{\sigma_v^2}{\left|1+a_1\cdot\mathrm{e}^{-j\omega}+a_2\cdot\mathrm{e}^{-j2\omega}\right|^2}$，$\omega\in[-\pi,\pi]$。

程序如下：

① 产生随机过程的样本值。

```
%program p506
a1=-0.195;
a2=0.95;
n=[1:1000]';
v0=randn(1,1000);
var=(1-a2)/(1+a2)*((1+a2).^2-a1.^2);
v=0.3*v0;
u0=[0 0];
num=1;
den=[1 a1 a2];
Zi=filtic(num,den,u0)
[u,Zf]=filter(num,den,v,Zi);
xlabel('n');ylabel('V0(n)');
title('White Noise');
legend('var=1',1)
subplot(211);
plot(n,v);
xlabel('n');ylabel('V(n)');
title('White Noise');
legend('var=0.3',1)
subplot(212);
plot(n,u);
xlabel('n');ylabel('X(n)');
title('signal with white noise');
```

运行结果如图 43-6 所示。

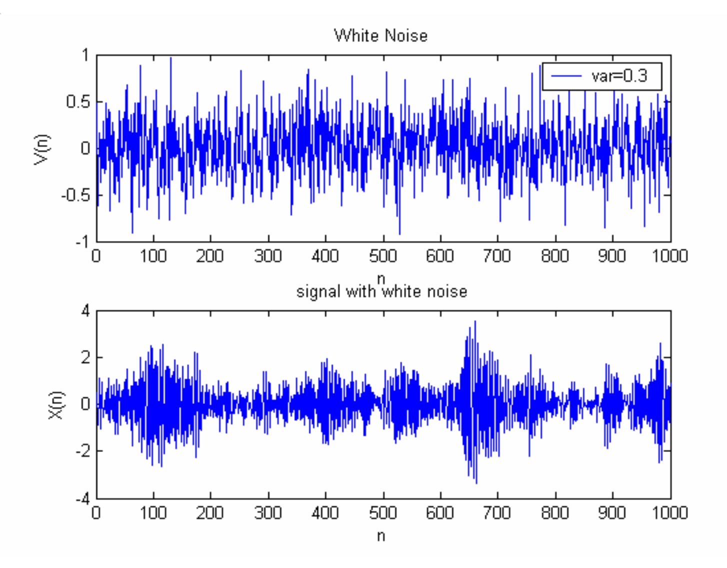


图 43-6　白噪声及含白噪声的信号

② 由公式 $r(m)=\frac{1}{n}\sum_{m=-\infty}^{\infty}u(n)u(n-m)$ 计算 $r\,(m)$, m =255

```
%program p507
lag=255;
[r,z]=xcorr(u,lag,'biased');
stem(z,r);
hndl=stem(z,r);
set(hndl,'Marker','o')
set(hndl,'MarkerSize',0.5)
xlabel('n');ylabel('r(n)');
title('Autocorrelation');
```

运行结果如图 43-7 所示。

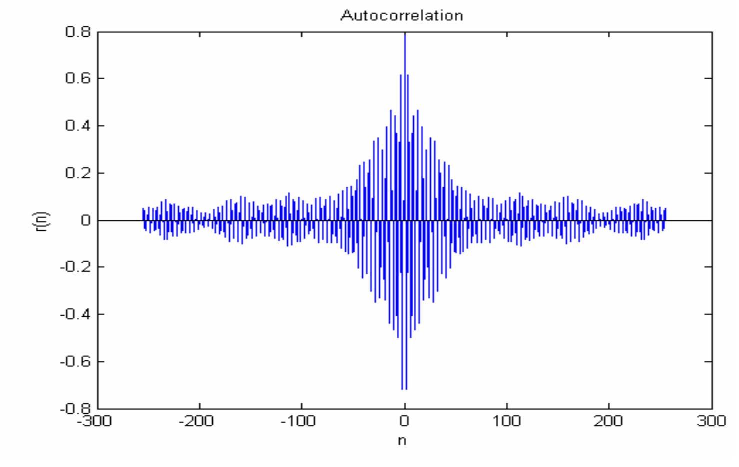


图 43-7　信号的自相关

③ 由 r(m)=−a1×r(m−1)−a2×r(m−2).计算得 a1, a2 的估计值为

$$\hat{a}_1=-0.1977,\ \hat{a}_2=0.9451$$

④ $P_{BT}(\omega)=\sum_{m=-255}^{255} r(m)\mathrm{e}^{-j\omega n}$

$$P_{AR}(\omega)=\frac{\sigma_v^2}{\left|1+a_1\cdot\mathrm{e}^{-j\omega}+a_2\cdot\mathrm{e}^{-j2\omega}\right|^2}$$

```
%program p508
%Black_turkey
m=1:255;
k=-500:500;
w=(pi/500)*k;
R=k/500;
r0=0.9141;
r2=r(257:511)
X=r2*cos(pi/500*m'*k)+r0;
magX=abs(X);
subplot(211);
plot(R,magX);
xlabel('Angular frequency,ω (π)');
ylabel('Pbt(ω)');
title('Black_Turkey 功率谱曲线')

%AR
HH=zeros(1,1001);
w=(pi/1000)*k;
a1=-0.195
a2=0.95;
var=1;

H=1+a1*exp(-j*pi/500*k)+a2*exp(-j*2*pi/500*k);
magH=abs(H);

i=1
while i<1001
   HH(i)=1/magH(i).^2;
   magH(i)=magH(i+1);
   i=i+1
end
HH(1)=1/magH(1).^2;
HH(1001)=1/magH(1001).^2;
subplot(212)
plot(R,var*HH)
xlabel('Angular frequency,ω (π)');
ylabel('Par(ω)');
title('AR 功率谱曲线')
```

运行结果如图 43-8 所示。

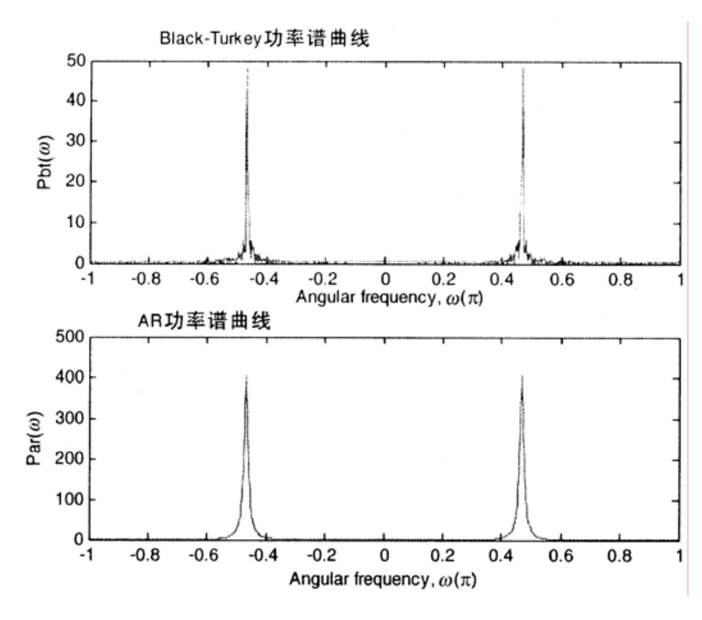


图 43-8　信号功率谱估计

43.4　基于特征分解功率谱估计方法

43.4.1　MUSIC 算法—Multiple Signal Classification(多信号分类法)

MUSIC 算法以自相关矩阵分析为基础，将系统自相关矩阵分为两个子空间：信号子空间和噪声子空间。为求功率谱估计，MUSIC 算法计算信号空间和噪声空间的特征值向量函数，使得在正弦信号频率处功率谱出现峰值，而其他地方函数最小。

MUSIC 算法计算步骤如下：

（1）根据 N 个观测样本值 $x_1, x_2, \cdots, x_n$ 估计 P+1 阶自相关矩阵 $\underline{R}$（实际估计 $\hat{r}(0), \hat{r}(1) \cdots \hat{r}(p)$）。

（2）对 $\underline{R}$ 进行特征分解，得到 P–M+1 个特征值，对应特征向量 $\underline{v}_{m+1} \cdots \underline{v}_{p+1}$

$$\underline{E}_N = l\{\underline{v}_{m+1} \cdots \underline{v}_{p+1}\}$$

（3）画出功率谱。

$$P_{\text{MUSIC}}(\omega) = \frac{1}{\sum_{i=M+1}^{p+1} \left| \underline{a}^H v_i \right|^2} = \frac{1}{\underline{a}^H \underline{E}_N \underline{E}_N{}^H a} = \frac{1}{\left\| \underline{a}^H \underline{E}_N \right\|^2}$$

在 $[0, 2\pi]$ 内改变 ω，画出谱线，找出峰值点。

【例 43-5】 $x(n) = \exp(j\pi n - j\pi) + \exp(j\omega_0 n - j0.7\pi)] + v(n)$ 为一复正弦加白噪声随机过程。$v(n)$ 为零均值白噪声。$\frac{S}{N} = 10\text{dB}$。

① 取 P=3，构造 4 阶的自相关矩阵 $\underline{R}$。

② 用 MUSIC 算法，估计功率谱。

MATLAB 程序如下：

```
%program p509
var=sqrt(1/exp(1.0));
v=var*randn(1,1000);

n=1:1000;
w0=1.4*pi;
xn=exp(j*pi*n-j*pi)+exp(j*w0*n-j*0.7*pi)+v;
m=-500:500;
[r,lag]=xcorr(xn,500,'biased');%corr

%pmusic
R=[r(501),r(502),r(503),r(504);
   r(500),r(501),r(502),r(503);
   r(499),r(500),r(501),r(502);
   r(498),r(499),r(500),r(501)];

[V,D]=eig(R);
V3=[V(1,3),V(2,3),V(3,3),V(4,3)].';
V4=[V(1,4),V(2,4),V(3,4),V(4,4)].';

p=0:3;
wm=[0:0.002*pi:2*pi];
B=[(exp(-j)).^(wm'*p)];
A=B.';

p1=A'*V3;
p2=A'*V4;
s=(abs(p1)).^2+(abs(p2)).^2;
pmus=1./s;

subplot(211);
pp=10*log10(pmus);
plot(wm/pi,pp);
ylabel('Power Spectrum Magnitude (dB)');
title(' Music Spectral Estimate')
```

%MATLAB 工具箱函数

```
subplot(212);
pmusic(xn,5)
```

运行结果如图 43-9 所示。

此处取 $\omega_0 = 1.4\pi$，可以改变 ω_0 的值来比较算法的性能。

MATLAB 工具箱提供了计算 MUSIC 功率谱的函数 pmusic。调用格式为：

$[P_{xx}, f] = \text{pmusic}(x, p)$

$[P_{xx}, f] = \text{pmusic}(x, [p, \text{thresh}])$

$[P_{xx}, f, a] = \text{pmusic}(x, [p, \text{thresh}], \text{Nfft}, F_s, \text{window}, \text{NoverLap})$

其中，x 为输入信号，可以是向量或者矩阵；p 为信号子空间维数；thresh 为阈值；Nfft 为采用的 FFT 长度，该值决定了功率谱估计速度；F_s 为采样频率；window 定义窗函数和 x 系列的长度，窗函数长度必须小于 Nfft，否则会出错；NoverLap 为分段序列重叠的采样点数。

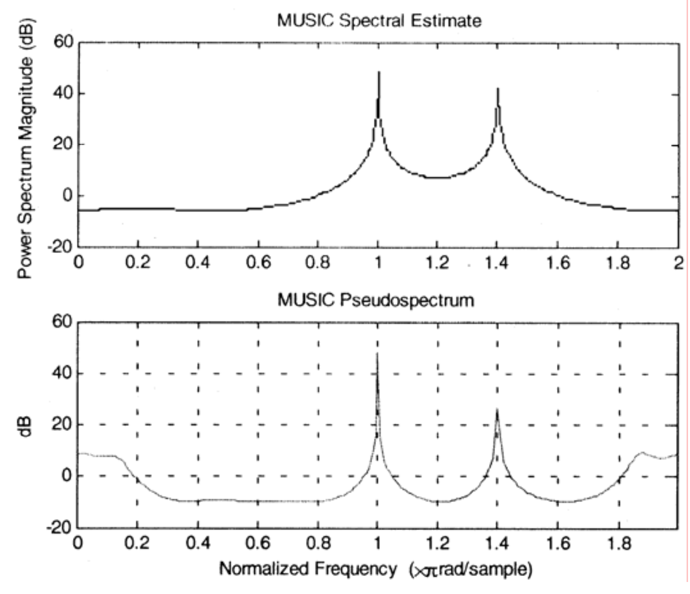


图 43-9　信号 MUSIC 功率谱估计

【例 43-6】　用 MUSIC 法估计信号 $x(t)\sin(2\pi\ f_1 t)+2\sin(2\pi\ f_2 t)+\omega(t)$ 的功率谱。其中，$f_1=50\text{Hz}$，$f_2=120\text{Hz}$，$\omega(t)$ 为白噪声，采样频率 $F_S=1000\,\text{Hz}$。

用 MATLAB 编程如下：

```
%program p510
N=1000;
n=0:N-1;
Fs=1024;
t=n/Fs;
xn=sin(2*pi*100*t)+2*sin(2*pi*200*t)+randn(1,N);
subplot(211);
pmusic(xn,7);
```

运行结果如图 43-10 所示。

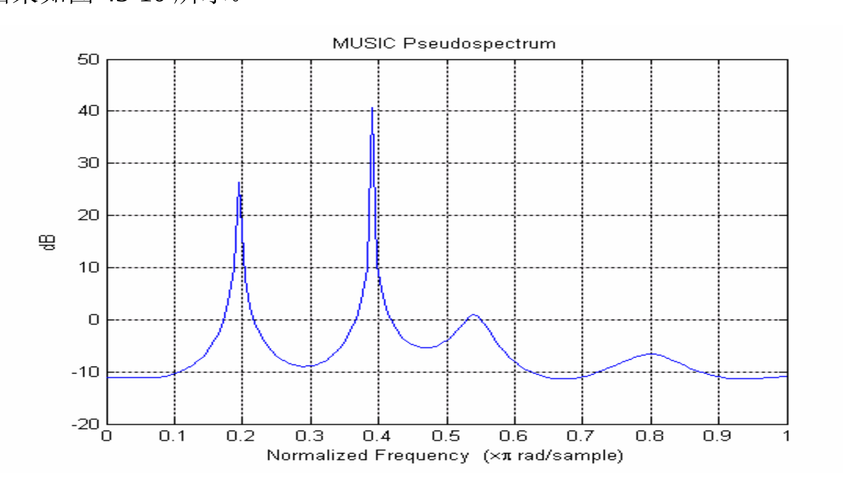


图 43-10　信号 MUSIC 功率谱估计

43.4.2 MVDR 算法—Minimum Varivance Distortionless Response (最小方差无失真响应)

MVDR 算法思想是使信号通过一 FIR 系统，而噪声尽量被抑制的方法。构造一代价函数，得到功率谱函数

$$P_{\text{MVDR}}(\omega)=\frac{1}{\underline{a}^H\underline{R}^{-1}\underline{a}}，\quad \underline{a}=[1,\mathrm{e}^{-j\omega},\mathrm{e}^{-j2\omega}\dots\mathrm{e}^{-jp\omega}]^T$$

在 $[0,2\pi]$ 内改变 ω，画出 $P_{\text{MVDR}}(\omega)$ 曲线，在 $\omega=\omega_t$ 处，会出现峰值。

MVDR 估计功率谱的步骤如下。

（1）由 $x(n)$ 的 N 个样本 $x(1),x(2)\cdots x(N)$ 估计 $\hat{r}(m)$，构造 P+1 阶 $\hat{\underline{R}}$；

（2）$P_{\text{MVDR}}(\omega)=\frac{1}{\underline{a}^H\underline{R}^{-1}\underline{a}}$，画曲线，找峰值位置。

【例 43-7】 $x(n)=\exp(j\pi n-j\pi)+\exp(j\omega_0 n-j0.7\pi+v(n))$ 为一复正弦加白噪声随机过程，$v(n)$ 为零均值白噪声。$S/N=10\text{dB}$。

用 MVDR 算法估计功率谱，$P_{\text{MVDR}}(\omega)=\frac{1}{\underline{a}^H\underline{R}^{-1}\underline{a}}$。

此处取 $\omega_0=1.4\pi$，可以改变 ω_0 的值来比较算法的性能。程序如下：

```
%program p511
var=sqrt(1/exp(1.0));
v=var*randn(1,1000);

n=1:1000;
w0=1.4*pi;
xn=exp(j*pi*n-j*pi)+exp(j*w0*n-j*0.7*pi)+v;
m=-500:500;
[r,lag]=xcorr(xn,500,'biased');%corr

R=[r(501),r(502),r(503),r(504);
   r(500),r(501),r(502),r(503);
   r(499),r(500),r(501),r(502);
   r(498),r(499),r(500),r(501)];

[V,D]=eig(R);
V3=[V(1,3),V(2,3),V(3,3),V(4,3)].';
V4=[V(1,4),V(2,4),V(3,4),V(4,4)].';

p=0:3;
wm=[0:0.002*pi:2*pi];
B=[(exp(-j)).^(wm'*p)];
A=B.';
%p-mvdr
z=A'*inv(R)*A;
Z=diag(z');
pmv=1./Z;
```

```
plot(wm/pi,pmv);
xlabel('Angular frequency,ω/pi');
ylabel('Power Spectrum Magnitude (dB)');
title(' MVDR Spectral Estimate')
```

运行结果如图 43-11 所示。

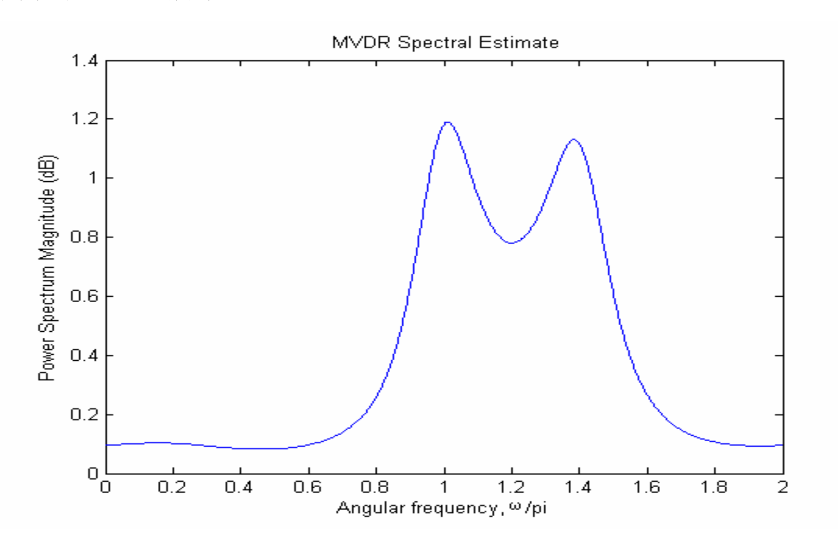


图 43-11　信号的 MVDR 功率谱估计

第六篇　曲线拟合工具箱

在实际工程应用和科学实践中，经常需要寻求两个（或多个）变量间的关系，而实际却只能测得一些分散的数据点。针对这些分散的数据点，运用某种拟合方法生成一条连续的曲线，这个过程称为曲线拟合。曲线拟合可分为参数拟合和非参数拟合。参数拟合采用的是最小二乘法，而非参数拟合采用的是插值法。

第 44 章　数据预处理

在曲线拟合之前必须对数据进行预处理，去除界外值、不定值和重复值，以减少人为误差，提高拟合的精度。数据预处理的内容包括两部分，即数据输入察看和数据的预处理。传输数据通过数据 GUI 来实现，察看数据点通过曲线拟合工具的散点图来实现。

44.1　输入数据集

44.1.1　打开曲线拟合工具界面

曲线拟合工具界面是一个可视化的图形界面，具有强大的图形拟合功能，包括：

① 可视化地展示一个或多个数据集，并可以用散点图来展示；

② 用残差和置信区间可视化地估计拟合结果的好坏；

③ 通过其他界面可以实现许多功能：输出、察看和平滑数据；拟合数据、比较拟合曲线和数据集；从拟合曲线中排除特殊的数据点；选定区间后，可以显示拟合曲线和数据集；还可以做内插法、外推法、微分或积分拟合。

可以通过 cftool 命令打开曲线拟合工具界面，如图 44-1 所示。

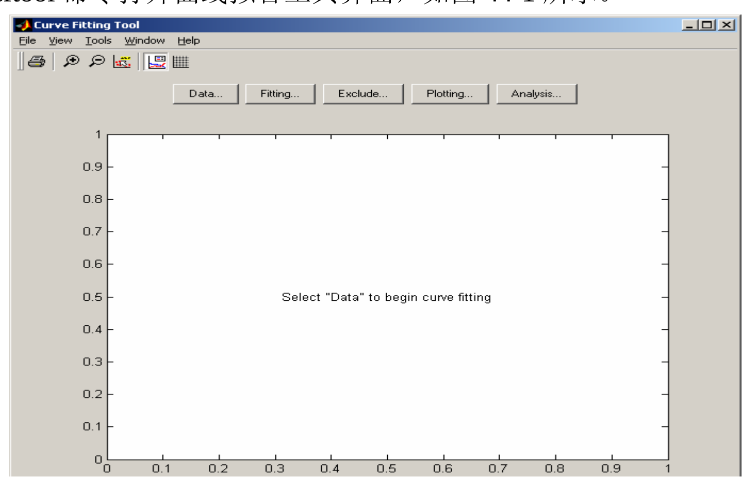


图 44-1　曲线拟合工具界面

曲线拟合工具中主要有 5 个命令按钮，即“Data”按钮、“Fitting”按钮、“Exclude”按钮、“Ploting”按钮和“Analysis”按钮。其功能如下：

- “Data”按钮　可输出、察看和平滑数据；
- “Fitting”按钮　可拟合数据、比较拟合曲线和数据集；
- “Exclude”按钮　可从拟合曲线中排除特殊的数据点；
- “Ploting”按钮　在选定区间后，单击按钮，可以显示拟合曲线和数据集；
- “Analysis”按钮　可以做内插法、外推法、微分或积分拟合。

44.1.2　输入数据集

在输入数据之前，数据变量必须存在于 MATLAB 的工作区间。可以通过 load 命令输入变量。单击曲线拟合工具界面中的“Data”按钮，打开“Data”对话框，如图 44-2 所示。在该对话框中进行设置，可以输入数据。

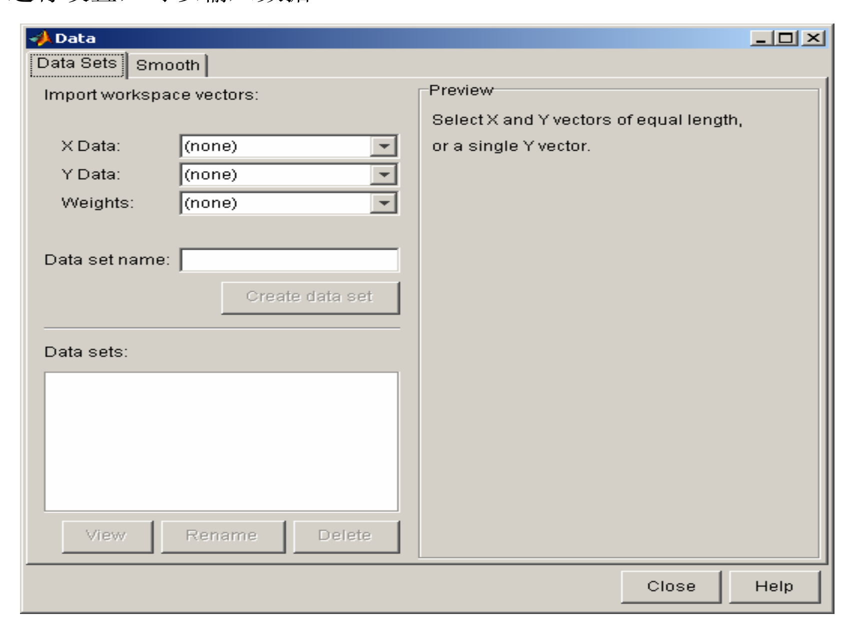


图 44-2　“Data”对话框的 Data Sets 选项卡

“Data”对话框包括两个选项卡：“Data Sets”选项卡和“Smooth”选项卡。

“Data Sets”选项卡中，各选项的作用如下：

- Import workspace vectors 把向量输入工作区。要注意的是变量必须具有相同的长度，无穷大的值和不定值被忽略。

　X data　用于选择预测数据。

　Y data　用于选择 X 的响应数据。

　Weight　用于选择权重，与响应数据相联系的向量，如果没选择，则默认值为 1。

- Preview 对所选向量进行图形化预览。
- Data set name 设置数据集的名称。工具箱可以随机产生惟一的文件名，但用户可以重命名，通过单击“Data set name”按钮并在打开的对话框中进行设置来实现。
- Data sets 选项以列表的形式显示所有拟合的数据集。当选择一个数据集时，可以对它

做如下操作：

- View 可以查看数据集，以图标形式和列表的形式，可以选择方法排除异常值；
- Rename 重命名；
- Delete 删去数据组。

【例 44-1】 输入数据，采用 MATLAB 自带的文件 census。census 有两个变量：cdate 和 pop。其中，cdat 是一个年向量，包括 1790~1990 年，间隔为 10 年；pop 是对应年份的美国人口。下面建立一个 M 文件：

```
load census
cftool(cdat, pop)
```

运行后产生如图 44-3 所示的对话框。单击“Data”按钮，显示如图 44-4 所示的对话框。

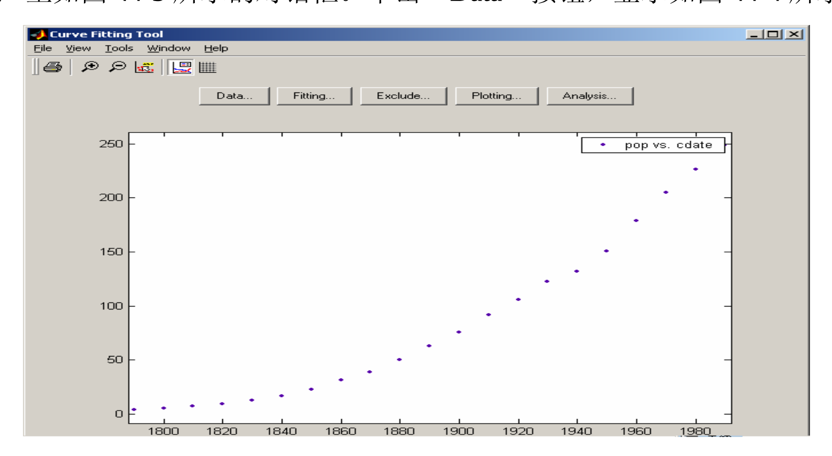


图 44-3 census 数据的散点图

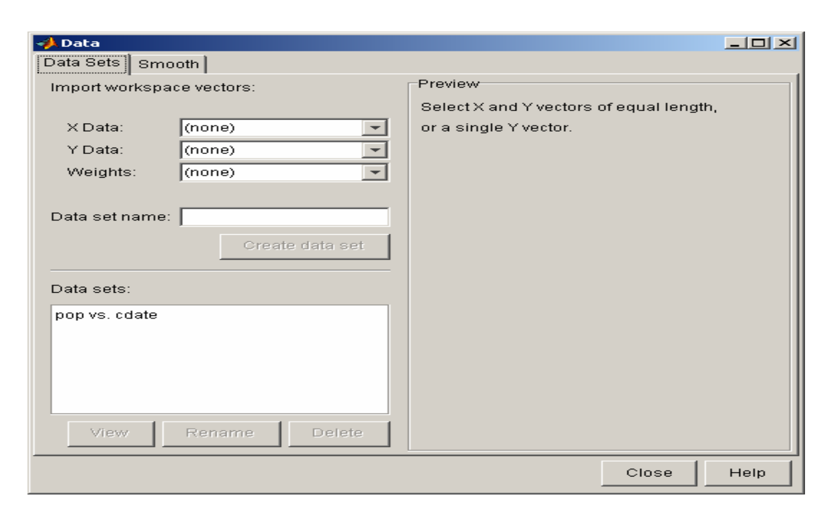


图 44-4 输入数据面板

在“X data”和“Y data”两个下拉式列表框中选择变量名，将在“Data”对话框中显示散点图的预览效果，如图 44-5 所示。

当选择“Data sets”列表框中的数据集时，单击“View”按钮，打开“View Data Set”对话框，如图 44-6 所示。

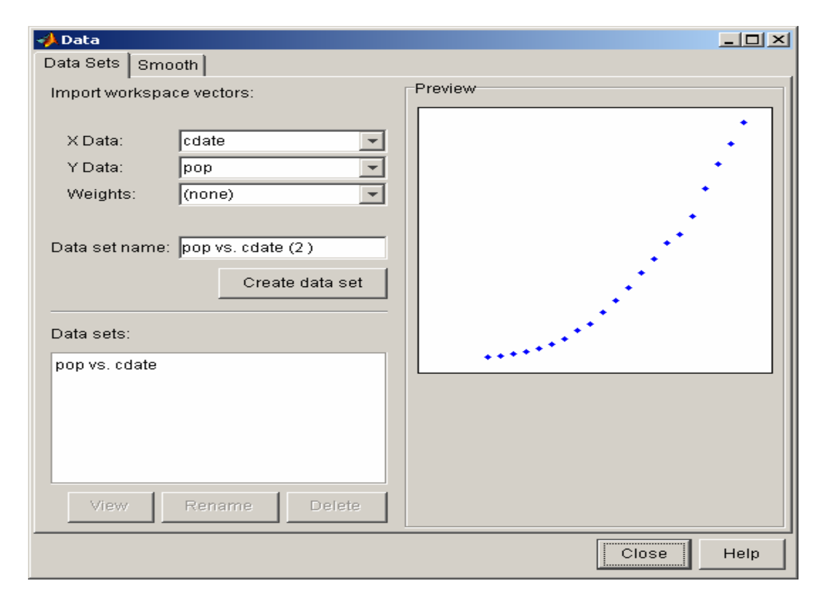


图 44-5　选择变量名存的预览效果

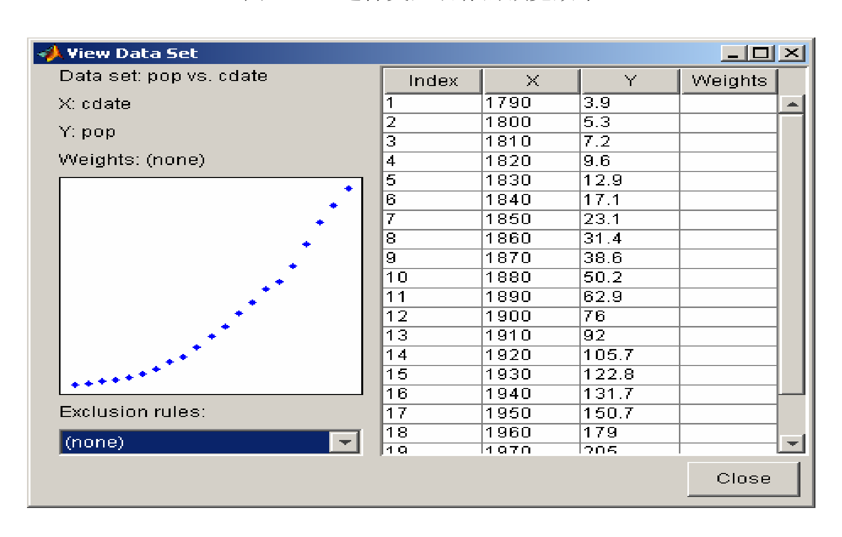


图 44-6　“View Data Set”对话框

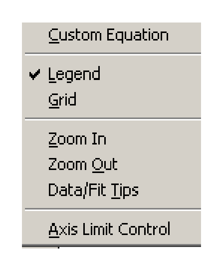
44.2　数据的查看

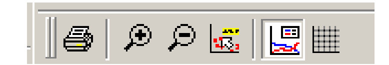
曲线拟合工具箱提供了两种数据查看方式，即散点图方式和工作表方式。

44.2.1　散点图方式

输入数据时，系统会自动以散点图的方式显示数据的分布情况。以预测数据为 Y 坐标，以响应数据为 X 坐标绘散点图。散点图具有强大的功能，它可以观测整个数据的分布趋势，然后可以根据该分布趋势考虑是否需要对数据进行预处理，以及采用什么样的拟合方法，这样有利于提高拟合的准确性。

曲线拟合工具箱提供多种方法编辑图形，包括利用图 44-7 的“Tool”主菜单、图 44-8 的 GUI 工具条和图 44-9 的右键快捷菜单等进行编辑。





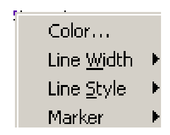


图 44-8　GUI 工具条　　　　图 44-9　右键快捷菜单　　　　图 44-7　“Tool”主菜单

图 44-10 显示了对拟合图形进行编辑的效果。

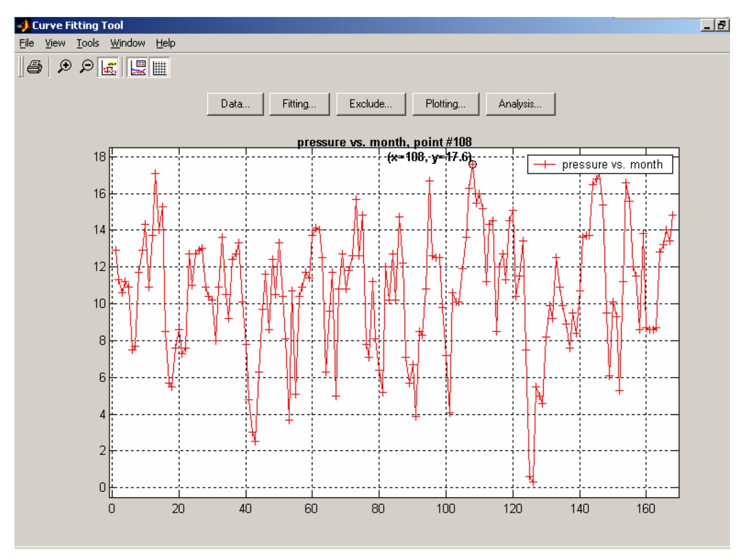


图 44-10　图形功能的显示

44.2.2　工作表方式

工作表方式如图 44-11 所示，在“View Data Set”对话框中，数据以电子表格的形式输出。在数据量比较小的情况下，通过这种方式，也可以看出一些数据的分布情况。在“View Data Set”对话框中，结合散点图和工作表两种方式，可以更有效地探察数据。

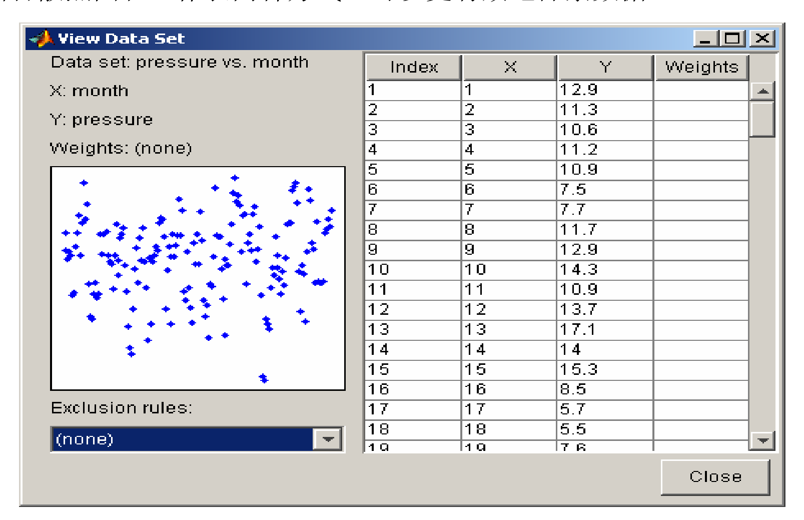


图 44-11　察看数据集的面板

44.3 数据的预处理

在曲线拟合工具箱中，数据的预处理方法主要包括平滑法、排除法和区间排除法等。

44.3.1 平滑数据

在处理数据时，数据有时杂乱无章，采用一些使图形更平滑的算法，可以提高以后的拟合效果。曲线的平滑需具备两个基本的假设条件：预测数据和响应数据之间本质上是平滑的；曲线平滑的结果是得到平滑的数据，它应该最能反映原始数据的本质特征。曲线平滑的结果是试图估计每个响应数据的均值分布。

注意：平滑数据后不能用参数模型拟合数据，因为数据平滑假设误差是符合正态分布的。

平滑数据是通过“Smooth”对话框来实现的。

打开曲线拟合工具箱，单击“Data”按钮，打开“Data”对话框，在其中选择“Smooth”选项卡，如图 44-12 所示。

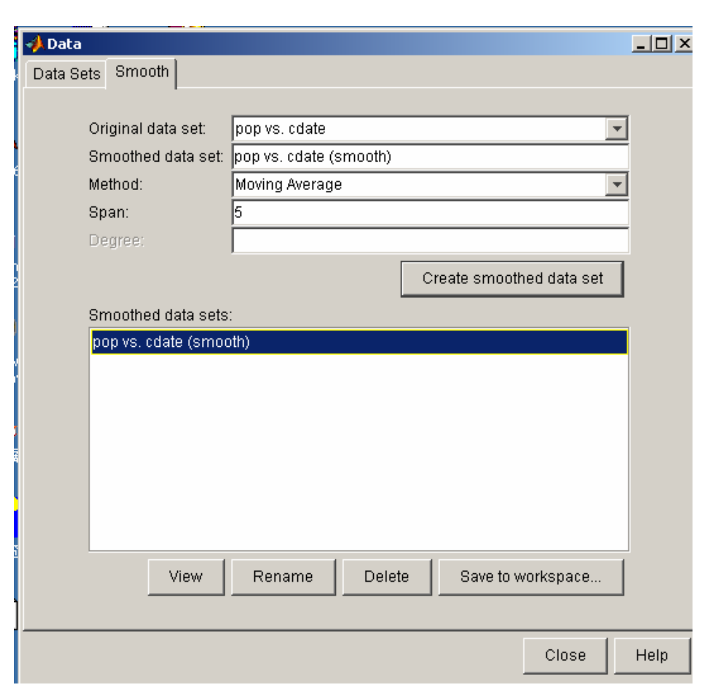


图 44-12 “Smooth”选项卡平滑数据面板

在“Smooth”选项卡中各选项的功能如下：

- Original data set 用于挑选需要拟合的数据集。
- Smoothed data set 平滑数据的名称。
- Method 用于选择平滑数据的方法，每一个响应数据用通过特殊的曲线平滑方法所计算的结果来取代。平滑数据的方法包括：
  - Moving average 用移动平均值进行替换；
  - Lowess 局部加权散点图平滑数据，采用线性最小二乘法和一阶多项式拟合得到的数

据进行替换；

➢ Loess　局部加权散点图平滑数据，采用线性最小二乘法和二阶多项式拟合得到的数据进行替换；

➢ Savitzky-Golay　采用未加权的线性最小二乘法过滤数据，利用指定阶数的多项式得到的数据进行替换；

➢ Span　用于进行平滑计算的数据点的数目；

➢ Degree　用于 Savitzky-Golay 方法拟合多项式的阶数。

● Smoothed data sets 对于所有平滑数据集进行列表。可以增加平滑数据集，通过单击“Create smoothed data set”按钮，可以创建经过平滑的数据集。

● View 按钮 打开查看数据集的 GUI，以散点图方式和工作表方式查看数据，可以选择排除异常值的方法。

● Rename 用于重命名。

● Delete 可删去数据组。

● Save to workspace 保存数据集。

【例 44-2】　用各种方法平滑数据并以图形形式输出。

```
x=[1 2 3 4 5 6 8];
y=[9 8 7 5 6 4 3];
plot(x,y);
cftool(x,y)
```

结果如图 44-13 所示。

图 44-13 显示了 7 种平滑曲线图，分别采用了不同的平滑方法，不同方法的平滑结果相差较大，根据需要读者可以选择所需的平滑结果。

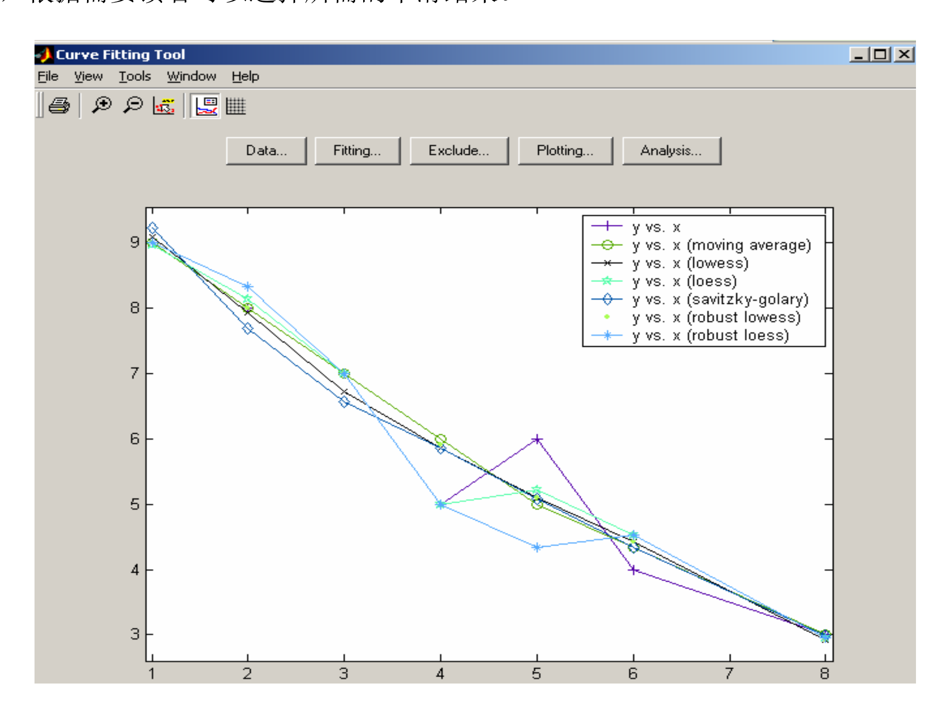


图 44-13　数据拟合图形

44.3.2　排除法和区间排除法

曲线拟合工具箱提供了两种排除数据的方法：排除法和区间排除法。排除法是对数据中的异常值进行排除。区间排除法是采用一定的区间去排除那些由于系统误差导致偏离正常值的异常值。

排除数据用“Exclude”对话框来实现，在曲线拟合工具中单击“Exclude”按钮，可以打开“Exclude”对话框，如图 44-14 所示。

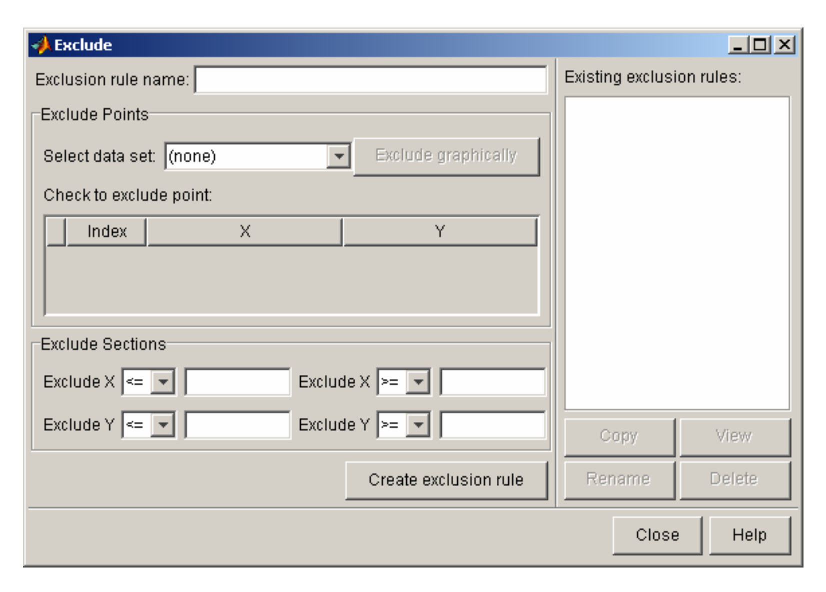


图 44-14　“Exclude”对话框

- Exclusion rule name 指定分离规则的名称。
- Existing exclusion rules 列表产生的文件名，当你选择一个文件名时，可以进行如下操作：
  - Copy　复制分离规则的文件；
  - Rename　重命名；
  - Relete　删去一个文件；
  - View　以图形的形式展示分离规则的文件。
- Select data set　挑选需要操作的数据集。
- Exclude graphically　允许你以图形的形式去除异常值，排除个别的点用“×”标记。
- Check to exclude point　挑选个别的点进行排除，可以通过在数据表中打勾来选择要排除的数据。
- Exclude Sections　选定区域排除数据。包括：
  - Exclude X　选择预测数据 X 要排除的数据范围；
  - ExcludeY　选择响应数据 Y 要排除的数据范围。

44.3.3　其他数据预处理方法

其他的预处理方法不便通过曲线拟合工具箱来完成，主要包括两部分：响应数据的转换

和去除无穷大、缺失值和异常值。

响应数据的转换一般包括对数转换、指数转换例如 $y^{1/2}$, y^{-1} 等等，用这些转换可以使非线形的模型线性化，便于曲线拟合。变量的转换一般在命令行里实现，然后把转换后的数据输入曲线拟合工具箱，进行拟合。

尽管无穷大、不定值在曲线拟合中可以忽略，但如果想把它们从数据集中删除，可以用 isinf 和 isnan 置换无穷大值和缺失值。

第 45 章　曲 线 拟 合

MATLAB 提供了两种方法进行曲线拟合。一种是以函数的形式，使用命令对数据进行拟合。这种方法比较繁琐，需要对拟合函数有比较好的了解。另外一种是用图形窗口进行操作，具有简便、快速，可操作性强的优点。MATLAB 提供了两种图形窗口，一种是基本的拟合界面，另一种是曲线拟合工具。基本拟合界面操作简单，可以做较为简单的曲线拟合，而曲线拟合工具功能强大，适用于各种复杂模型的曲线拟合。

45.1　有关函数介绍

45.1.1　多项式拟合函数

1．Polyfit 函数

利用该函数进行多项式曲线拟合，其调用格式为：

● p = polyfit(x, y, n)用最小二乘法对数据进行拟合，返回 n 次多项式的系数，并用降序排列的向量表示，长度为 n+1。

$$P(x) = P_1X^n + P_2X^{n-1} + P_3X_{n-2} + \cdots + P_nX + P_{n+1}$$

● [p, S] = polyfit(x, y, n)返回多项式系数向量 p 和矩阵 S。S 与 polyval 函数一起用时，可以得到预测值的误差估计。若数据 y 的误差服以方差为常数的独立正态分布，则 polyval 函数将生成一个误差范围，其中包含至少 50%的预测值。

● [p, S, mu] = polyfit(x, y, n)返回多项式的系数，mu 是一个二维向量[μ_1　μ_2]，μ_1＝mean(x)，μ_2=std(x)，对数据进行预处理 $x = (x-\mu_1)/\mu_2$。

2．Polyval 函数

利用该函数进行多项式曲线拟合评价，其调用格式为：

● y = polyval(p, x)返回 n 阶多项式在 x 处的值，x 可以是一个矩阵或者是一个向量，向量 p 是 n+1 个以降序排列的多项式的系数。即

$$P(x) = P_1X^n + P_2X^{n-1} + P_3X_{n-2} + \cdots + P_nX + P_{n+1}$$

● y = polyval(p, x, [ ], mu)用 $x=(x-\mu_1)/\mu_2$ 代替 x，其中 μ_1＝mean(x), μ_2=std(x)。mu=[μ_1, μ_2]，通过这样处理数据，使数据合理化。

● [y, delta] = polyval(p, x, S) 和 [y, delta] = polyval(p, x, S, mu)产生置信区间 y±delta。如果误差结果服从标准正态分布，则实测数据落在 y±delta 区间内的概率至少为 50%。

【例 45-1】 根据下面给定的数据进行多项式拟合。

```
x=[0 0.0385 0.0963 0.1925 0.2888 0.385];
y=[0.042 0.104 0.186 0.338 0.479 0.612];
[p,s]=(polyfit(x,y,5));
[p,s,mu]=(polyfit(x,y,5))
```

则结果为

```
p =
      0.0193   -0.0110   -0.0430    0.0073    0.2449    0.2961
s =
        R: [6x6 double]
       df: 0
normr: 1.2039e-016
```

则拟合的多项式为

$$P = 0.0193X^5 - 0.0110X^4 - 0.043X^3 + 0.0073X^2 + 0.2449X + 0.2961$$

自由度为 0，标准偏差为：1.2039e-016。

【例 45-2】 根据表 45-1 中的数据进行 4 阶多项式拟合。

表 45-1 数据表

X	1	3	4	5	6	7	8	9	10
F(x)	10	5	4	2	1	1	2	3	4

在命令窗口键入：

```
x=[ 1 3 4 5 6 7 8 9 10];
y=[10 5 4 2 1 1 2 3 4]
[p,s]=polyfit(x,y,4);
y1=polyval(p,x);
polt(x,y,'go',x,y1,'b--')
```

拟合结果如图 45-1 所示。

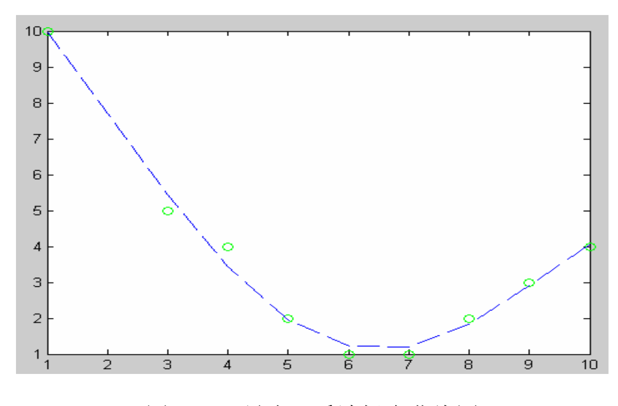


图 45-1 最小二乘法拟合曲线图

【例 45-3】 计算多项式 $P(x)=3x^2+2x+1$ 在 $x=5, 7, 9$ 上的值。

在命令窗口键入：

```
p = [3 2 1];
polyval(p,[5 7 9])
```

结果为：

```
ans =
    86   162   262
```

45.1.2 其他函数

其他函数包括进行数据处理的函数、提供帮助信息的函数和设置模型的函数等。各函数

的功能如表 45-2 所示。读者若要具体了解各函数的功能，可以参见帮助文档。

表 45-2　数据拟合函数表

函　　数	功 能 描 述
cfit	产生拟合的目标
fit	用库模型、自定义模型、平滑样条或内插方法来拟合数据
fitoptions	产生或修改拟合选项
fittype	产生目标的拟合形式
cflibhelp	显示一些信息，包括库模型、三次样条和内插方法等
disp	显示曲线拟合工具的信息
get	返回拟合曲线的属性
set	对于拟合曲线显示属性值
excludedata	指定不参与拟合的数据
smooth	平滑响应数据
confint	计算拟合系数估计值的置信区间边界
differentiate	对于拟合结果求微分
integrate	对于拟合结果求积分
predint	对于新的观察量计算预测区间的边界
cftool	打开曲线拟合工具
datastates	返回数据的描述统计量
feval	估计一个拟合结果或拟合类型
polt	画出数据点、拟合线、预测区间、异常值点和残差

45.2　曲线的参数拟合

具体进行曲线拟合时，需要首先应用图 44-2 所示的“Data”对话框指定要分析的数据，然后在图 44-1 所示的“Curve Fitting Tool”对话框中单击“Fitting”按钮，打开“Fitting”对话框，如图 45-2 所示。在该对话框中进行设置，可以实现曲线拟合。

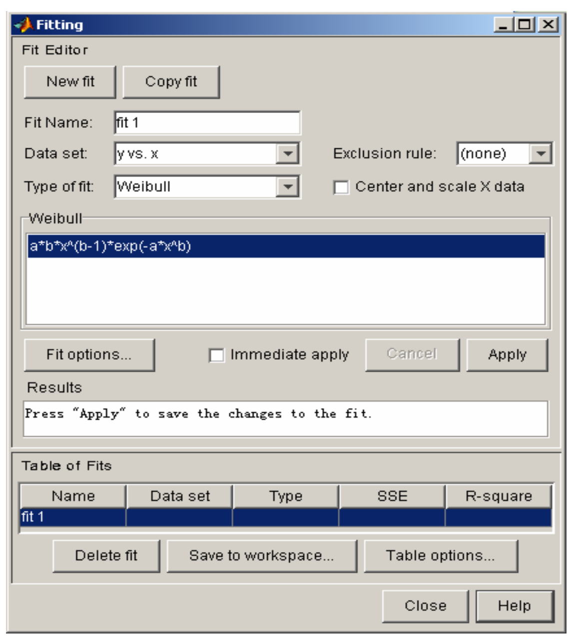


图 45-2　“Fitting”对话框

“Fitting”对话框包括两个面板：“Fit Editor”面板和“Table of Fits”面板。

① Fit Editor 选择拟合的文件名、数据集，选择排除数据的文件，比较数据拟合的各种方法，包括库函数、自定义的拟合模型和拟合参数的选择。

② Table of Fits 同时列出所有的拟合结果。

下面对上述两个面板分别进行详细描述：

● “New fit”和“Copy fit”按钮　开始进行曲线拟合时，单击“New fit”按钮，它采用默认的线性多项式拟合数据。在原有的拟合形式上，选择不同的曲线拟合方法，可以用“Copy fit”按钮。

● Fit name 选项为当前拟合曲线的名字。单击“New fit”按钮时系统会产生默认的文件名。

● Data set 选项为当前的数据集。

● Exclusion rule 排除异常值的文件名，在数据预处理前建立的文件名。

● Center and scale X data 可对预测数据进行中心化和离散化处理。

● Type of fit，拟合的类型，包括参数拟合和非参数拟合两种。具体包括：

- Custom Equations　自定义拟合的线性或非线性方程；
- New equation　使用“Custom Equations”按钮前，必须先单击“New equation”按钮选择合适的方程；
- Exponential　指数拟合包括两种形式：

  y=a*exp(b*x),

  y= a*exp(b*x)+c*exp(d*x)；
- Fourier　傅立叶拟合，正弦与余弦之和（共 8 个多项式）：

  a_0 +a_1*cos(x*w)+b1sin(x*w)

  a_0 +a_1*cos(x*w)+b1sin(x*w)+ a_2*cos(2x*w)+$b_2$sin(2x*w)

  ⋮

  a_0 +a_1*cos(x*w)+b1sin(x*w)⋯a_8*cos(8x*w)+$b_8$sin(8x*w)
- Gaussian　高斯法，包括 8 个公式：

  a_1*exp(-((x-b1)/c1)^2)

  ⋮

  a_1*exp(-((x-b1)/c1)^2)⋯⋯a_8*exp(-((x-b_8)/c_8)^2)
- Interpolant　内插法，包括线性内插、最近邻内插、三次样条内插和 shape-preserving 内插；
- Polynomial　多项式，从一阶至九阶；
- Rational　有理拟合，两个多项式之比，分子与分母都是多项式；

$$y = \frac{p_1 x^n + p_2 x^{n-1} + ... + p_{n+1}}{x^m + q_1 x^{m-1} + ... + q_m}$$

- Power　指数拟合，包括两种形式：y=a*x^b;　　y=a*x^b+c；
- Smoothing spline　平滑样条拟合，默认的平滑参数由拟合的数据集来决定，参数是 0 产生一个分段的线形多项式拟合，参数是 1 产生一个分段三次多项式拟合；
- Sum of Sin Functions　正弦函数的和，采用以下 8 个公式：

  a1*sin(b1*x+c1)

⋮

a1*sin(b1*x+c1)+⋯⋯+ a8*sin(b8*x+c8)

- Weibull　两个参数的 Weibull 分布，表达式如下：

  Y=a*b*x^(b-1)*exp(-a*x^b)

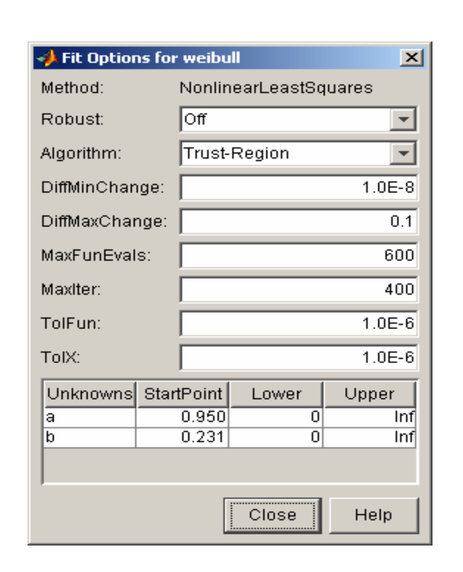


图 45-3　“Fit Options for weibull”对话框

● Fit options　包括一些拟合方法，如线性拟合、非线性拟合，以及其他选项，如图 45-3 所示。

“Apply”和“Immediate apply”按钮：单击“Apply”按钮，采用上述所选各种方法进行拟合；单击“Immediate apply”按钮，在选择一个拟合形式后立即输出结果并存储。

● Results 罗列进行拟合的各种参数，包括拟合的形式、置信区间大于 95%时的相关系数以及显示拟合效果好坏的各种参数，包括：

- *SSE*-sum of squares due to error　误差平方和，越接近于 0 曲线的拟合效果越好。

$$SSE = \sum_{i=1}^{n} W_i (Y_i - \hat{y}i)^2$$

R-Square-*SSR* 与 *SST* 的比值，*SSR* 与 *SST* 的定义为

$$SSR = \sum_{i=1}^{n} W_i (\hat{y}i - \overline{Y})^2$$

$$SST = \sum_{i=1}^{n} W_i (Y_i - \overline{Y})^2$$

$$R - \text{square} = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$

R-Square 的取值是 0~1，数值越接近 1，曲线的拟合效果越好。

- Degree of Freedom Adjusted R-Square　调整自由度以后的残差的平方，自由度为响应数据的个数 n 减去被拟合的相关系数的个数 m，即

$$v=n-m$$

Adjusted R-Square 的数值越接近 1，曲线的拟合效果越好。

- Root Mean Square Error　根的均方误差，由 RMSE 表示：

$$RMSE = S = \sqrt{MSE}$$

$$MSE = \frac{SSE}{V}$$

MSE 越接近 0，曲线的拟合效果越好。

- Table of fits　拟合曲线的列表，可以对每个列表作如下操作：

  Delete fit　删除所选的拟合曲线；

  Save to workspace　可以储存所有的拟合信息；

  Table options　选择与拟合相联系的信息。

【例 45-4】　用三次多项式和五次多项式拟合下列数据，并比较两种拟合的效果。

```
rand('state', 0)
x = [1: 0.1: 3 9: 0.1: 10]';
c = [2.5  -0.5  1.3  -0.1];
y =  c(1) + c(2)*x + c(3)*x.^2 + c(4)*x.^3 + (rand(size(x))-0.5);
cftool(x, y);
```

建立一个 M 文件，运行上述文件，打开曲线拟合工具，在数据栏里挑选数据，如图 45-4 所示。

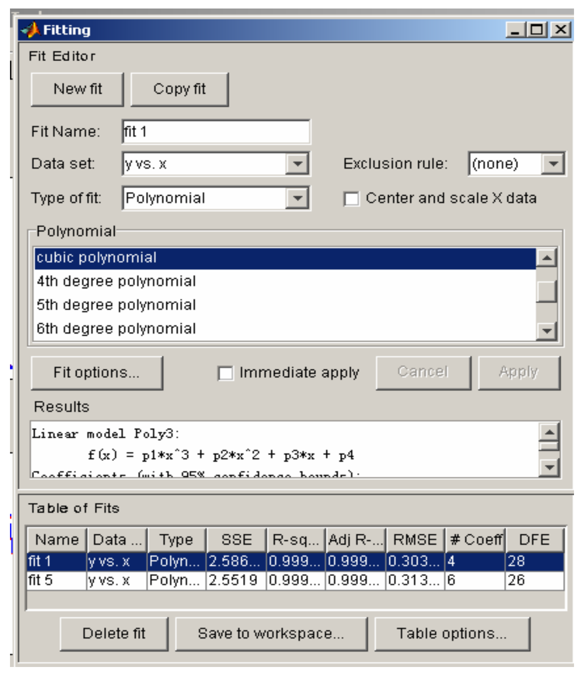


图 45-4　“Fitting”对话框

拟合的图形如图 45-5 所示。

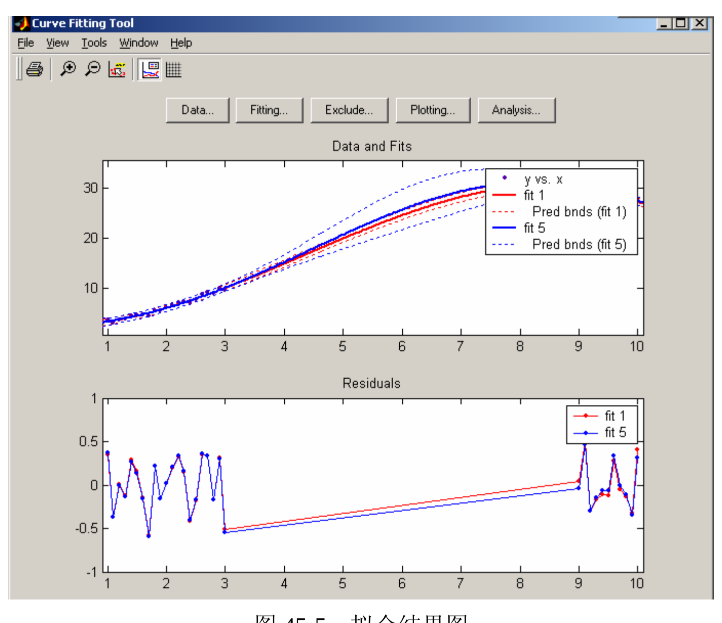


图 45-5　拟合结果图

图 45-5 中，fit1 曲线对应三次多项式的拟合结果和预测区间。fit5 曲线对应五次多项式的拟合结果和预测区间。由图可知三次多项式的拟合效果最好，五次多项式的预测区间较宽，效果不好，p4, p5, p6 的预测区间均较宽。这一点，从拟合结果的参数也可看出，如图 45-6 和图 45-7 所示。

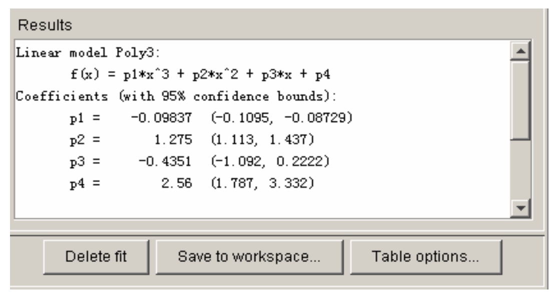


图 45-6 三次多项式的拟合结果

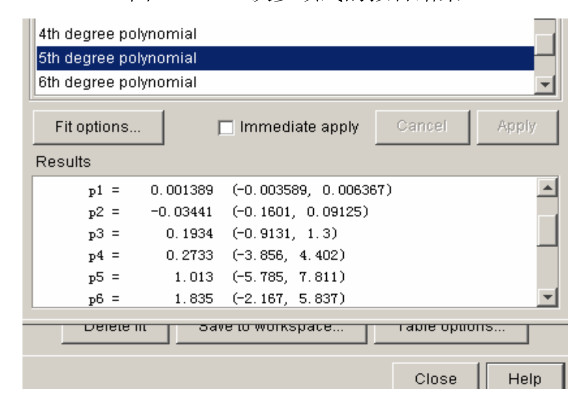


图 45-7 五次多项式的拟合结果

【例 45-5】 下面用有理拟合方法拟合数据 hahn1.m。hahn1.m 由 MATLAB 自带，描述铜的热膨胀与热力学温度的相关性，包括两个向量 tep 与 thermex。

```
load hahn1
Cftool(temp, thermex)
```

输入数据，打开曲线拟合工具对话框，如图 45-8 所示。

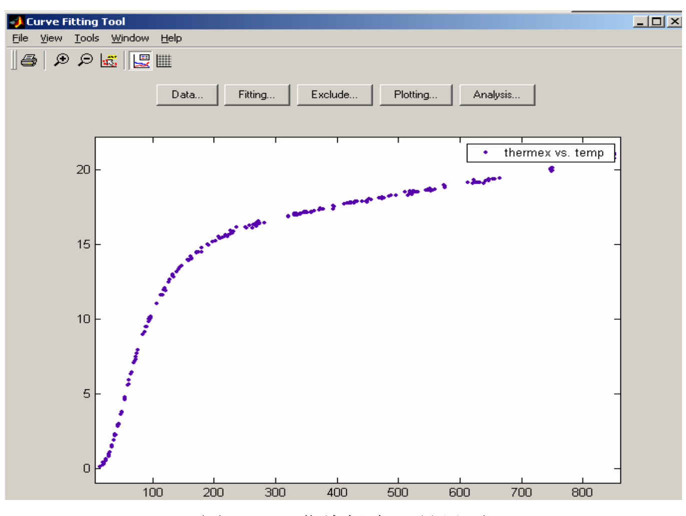


图 45-8 曲线拟合工具界面

单击“Fitting”按钮，选择曲线拟合类型“Rational”，先选择分子、分母均为二次多项式，比较拟合效果，如图 45-9、45-10 所示。

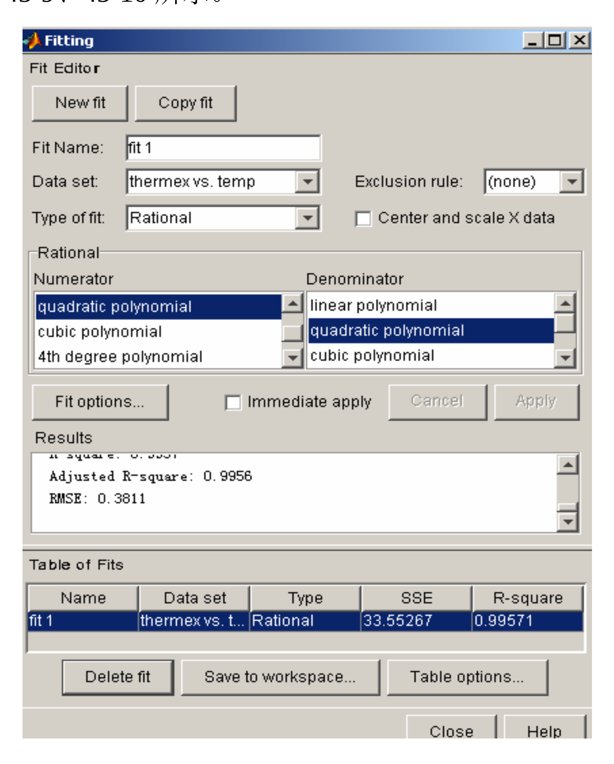


图 45-9　拟合界面

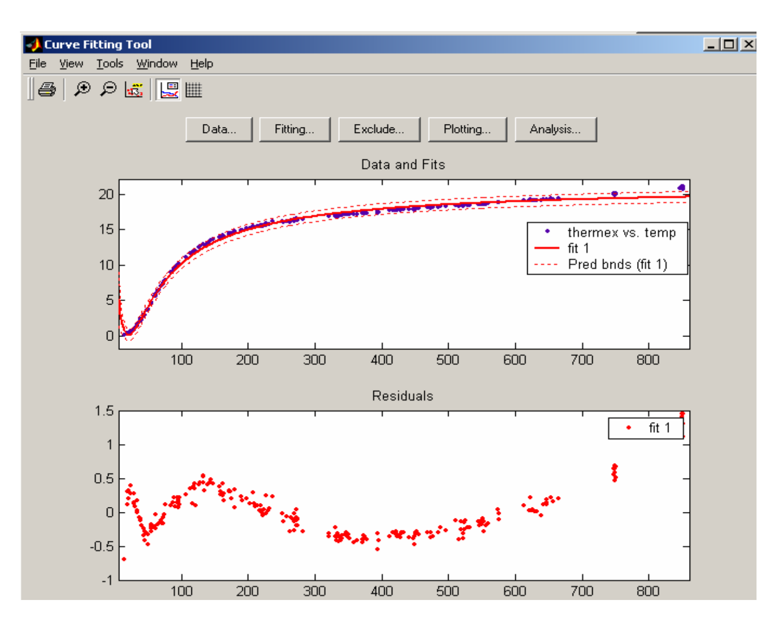


图 45-10　拟合结果

二次多项式拟合的结果明显地遗漏了最小值和大量的预测数据。此外，从残差点的分布还看得见明显的线形特征，这说明还存在更好的拟合模型。

选择分子与分母都是三次多项式的方程进行拟合，结果如图 45-11 所示。

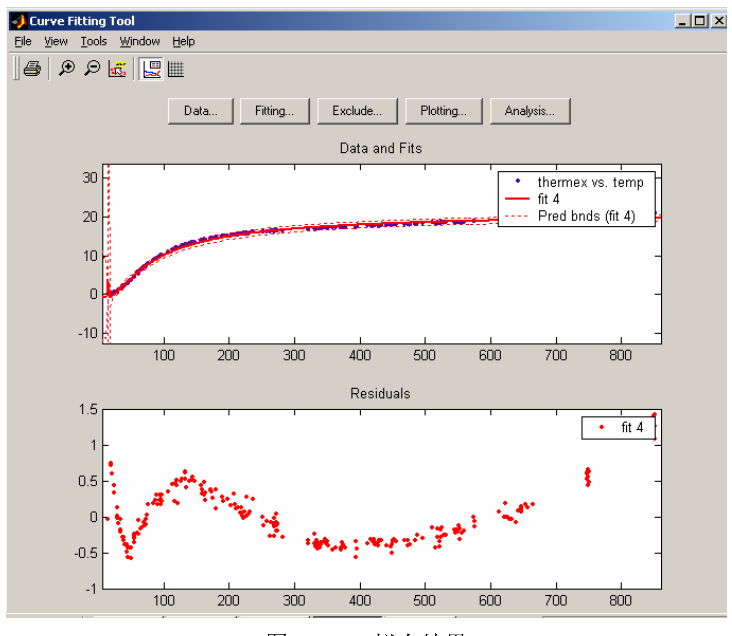


图 45-11　拟合结果

从图 45-11 中可以看出，采用这种模型得到的拟合曲线上有几个断点。它们是因为分母为零时函数溢出造成的。所以这个模型也不够好。

下面运用分子是三次多项式，分母是二次多项式的方程进行拟合，效果如图 45-12 所示。拟合效果很好，拟合曲线充分体现了整个数据，残差随机分布在 0 附近，可以选择。

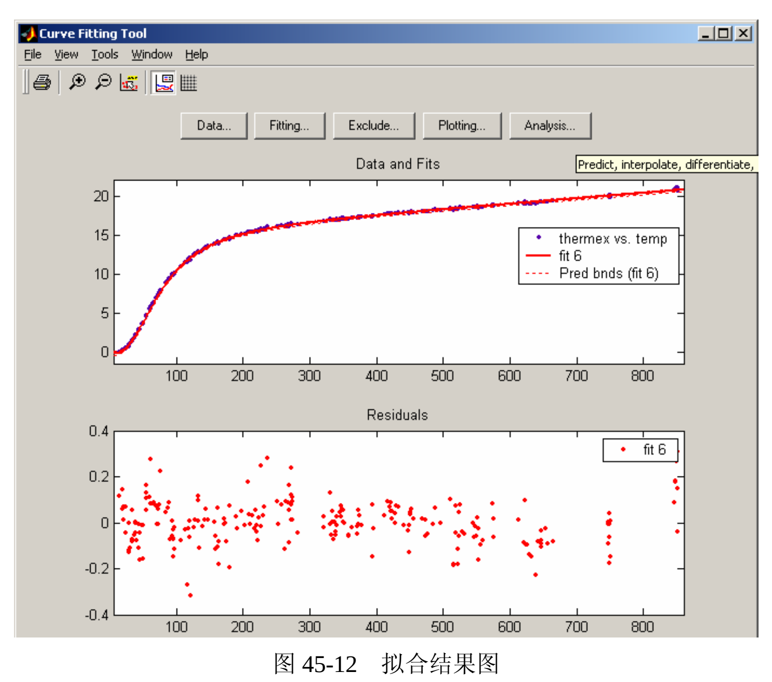


图 45-12　拟合结果图

45.3　非参数拟合

有时我们对拟合参数的提取或解释不感兴趣，只想得到一个平滑的通过各数据点的曲线，

这种拟合曲线的形式称之为非参数拟合。非参数拟合的方法包括内插（Interpolants）法，或称插值法和平滑样条（Smoothing spline）内插法。

内插法是在已知数据点之间估计数值的过程，包括如表 45-3 所示的方法。

表 45-3　内插方法

方　　法	描　　述
linear	线性内插，在每一队数据之间用不同的线性多项式拟合
Nearest neighbor	最近邻内插，内插点在最相邻的数据点之间
Cubic spline	三次样条内插，在每一队数据之间用不同的三次多项式拟合
Shape-preserving	分段三次艾尔米特内插（PCHIP）

平滑样条内插法是对杂乱无章的数据进行平滑处理，可以用平滑数据的方法来拟合，平滑的方法在数据的预处理中已经介绍。

【例 45-6】　用内插法拟合 carbon12alpha.mat 数据，包括最近邻内插法和 PCHIP 内插拟合法。

建立一个 M 文件，打开曲线拟合对话框，分别选择最近邻内插法和 PCHIP 内插法，拟合结果如图 45-13 所示。

```
load carbon12alpha
cftool(counts, angle)
```

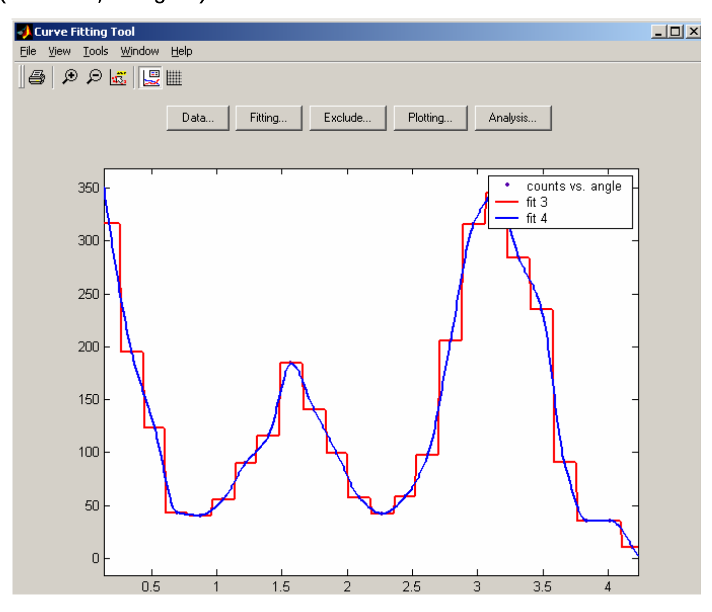


图 45-13　两种拟合结果图

图 45-13 中的 fit3 曲线是运用最近邻内插法得到的拟合曲线，fit4 曲线是运用 PCHIP 内插拟合法得到的拟合曲线。两种方法差别较大，具有不同的用途。fit3 曲线没有沿着数据点拟合，得出一个像阶梯状的图形。如果不考虑曲线的物理意义，则拟合结果应考虑 fit4 曲线。

【例 45-7】　用三次样条内插和集中平滑样条内插法拟合下列数据。

```
rand('state',0);
x = (4*pi)*[0 1 rand(1,25)];
y = sin(x) + .2*(rand(size(x))-.5);
```

结果如图 45-14 和图 45-15 所示，上述参数不能指示内插拟合的好坏。

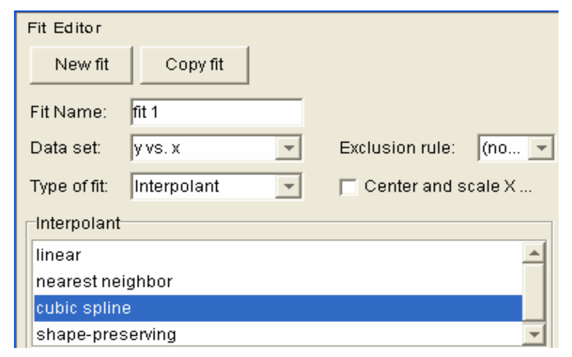


图 45-14　三次样条内插的选择

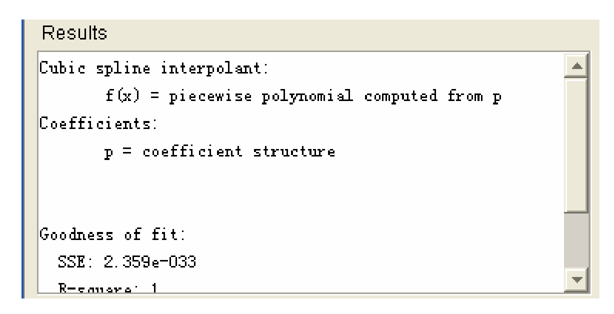


图 45-15　三次样条内插的拟合结果

下面用默认的平滑参数 P 和给定的平滑参数 0.5 分别进行平滑内插拟合，图 45-16 和图 45-17 为默认条件下的设置界面和拟合结果。图 45-18 显示了平滑参数为默认设置和为 0.5 时对应的拟合曲线。

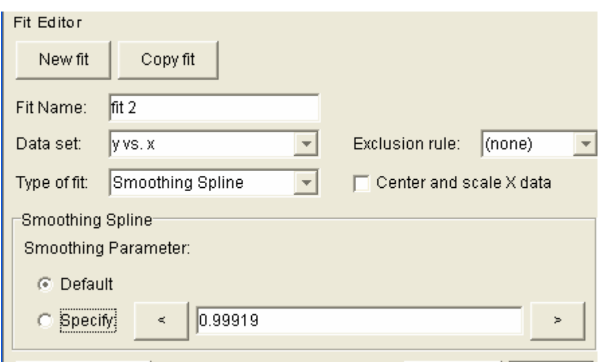


图 45-16　默认条件下的拟合设置界面

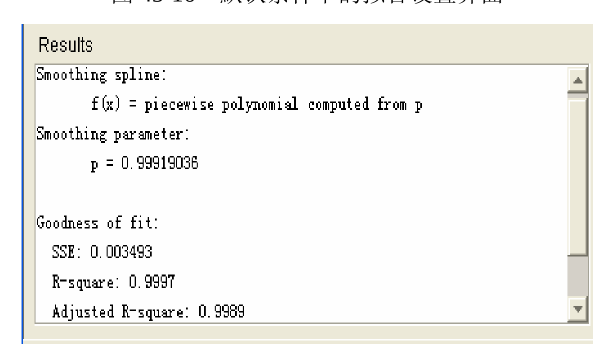


图 45-17　默认条件下的拟合结果

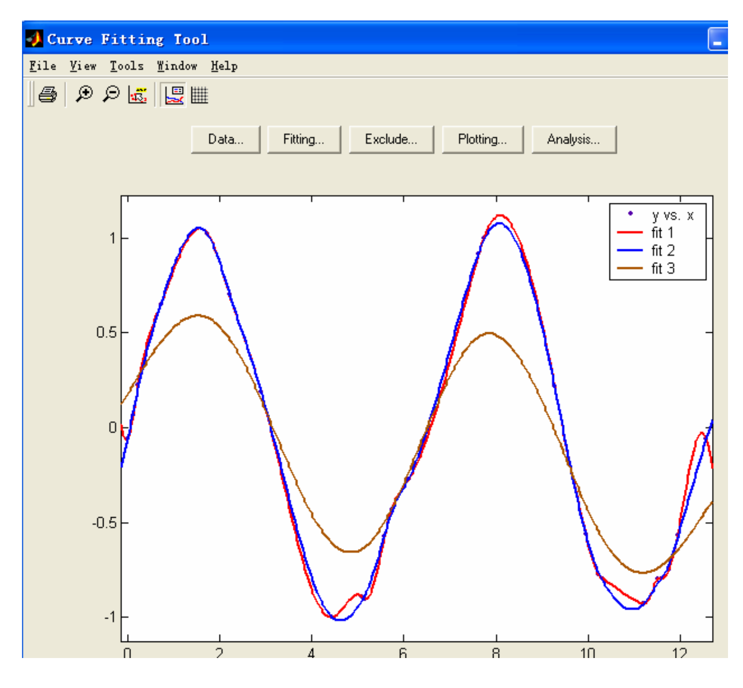


图 45-18　三种拟合方法的拟合结果图

曲线的平滑级别用图 45-16 中的“Smoothing Parameter”选项给定。默认的平滑参数值与数据集有关，并在单击“Apply”按钮以后由工具箱自动计算。对于本数据集，默认的平滑参数值接近于 1，表示平滑样条接近于三次样条，并且几乎正好穿过每个数据点。如果不想使用默认拟合参数生成的平滑级别，可以自己指定一个 0 和 1 之间的值。值为 0 时，生成一个分段线性多项式的拟合；值为 1 时，生成一个分段三次多项式的拟合，它穿过所有的数据点。

图 45-18 对平滑参数不同时的几个拟合结果进行了比较。其中，fit1 曲线是三次样条内插拟合结果图，fit2 曲线是在默认的平滑参数下的平滑样条内插拟合结果，fit3 曲线是在平滑参数为 0.5 时的平滑样条内插拟合结果。在默认的平滑参数下，拟合结果最好，平滑参数设置为 0.5 时，拟合结果最差。

45.4　基本的拟合界面

MATLAB 除了提供上述拟合的工具箱以外，还提供了一个方便简洁的拟合界面。它具有拟合快速，操作简便的优势，但拟合方法较少。

该界面具有如下功能。

- 数据的拟合可以通过内插法、分段三次艾尔米内插（PCHIP）或者是 1 到 10 阶的多项式实现。
- 利用一组数据可以同时做多个拟合曲线。
- 可以绘制残差图。
- 可以检查拟合结果。
- 可以进行拟合曲线的估计。
- 可用拟合结果和标准参差来注释。
- 可以保存拟合曲线和计算结果。

用户可以把该界面与命令函数结合起来应用。

按照下面的步骤打开曲线拟合界面。

（1）导入数据。

（2）在 Tools 菜单中单击 Basic Fitting 菜单选项。

（3）双击按钮。打开的曲线拟合界面“Basic Fitting”对话框，如图 45-19 所示。

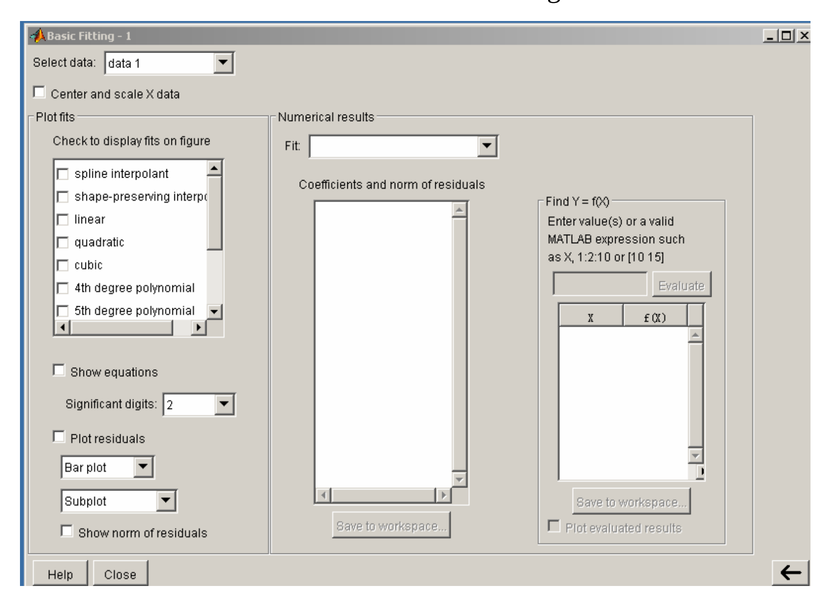


图 45-19　基本的拟合界面“Basic Fitting”对话框

“Basic Fitting”对话框中各选项的功能包括：

“Select data”下拉式列表框。在该列表框中挑选所要拟合的数据，一次只能选择一组数据，但在一组数据里可以同时拟合多条曲线。可以用“Plot Editor”改变数据的名称。

“Center and scale x data”单选框，可对数据进行中心化和离散化，数据经过处理后可以提高计算准确性。

“Plot fits”方框用于拟合图形。在当前的数据下，这个子菜单里给用户提供了多种拟合方法。

- Check to display fits on figure 列表框 提供了多种曲线拟合的方法，用户可以从中选择一个或多个。计算以后，选中的各个方法所对应的拟合曲线显示在图形界面中，可以根据标准残差的大小来选择理想的拟合曲线。一般标准残差越小，曲线的拟合效果越好。
- Show equation 单选钮：选此项，在拟合图形上显示方程。
- Plot residuals 单选钮：选此项，显示拟合曲线的残差，可用柱状图、散点图或线形图显示。
- Show norm of residuals 单选钮：选此项，显示标准残差。

“Numerical results”方框用于计算当前拟合图形的参数。

- Fit：选择当前数据集拟合的方程。
- Coefficients and norm of residuals：显示拟合数据集的计算结果。
- Save to workspace：保存计算结果。

➢ Find Y = f(X)：用内插法或外推法求当前拟合曲线的值。

【例 45-8】　用基本拟合界面拟合 MATLAB 软件自带数据 census.mat。

首先建立一个 M 文件：

```
load census
plot(cdate, pop, 'ro)
```

运行上述文件，得到图 45-20。单击“Tools”菜单，选择“Basic Fitting”菜单项拟合图中的点。“Basic Fitting”界面如图 45-21 所示。在该对话框中选择预处理数据，选择拟合方法 cubic 并画出残差图，列出标准残差，同时可以外推数据点，如图 45-22 所示。

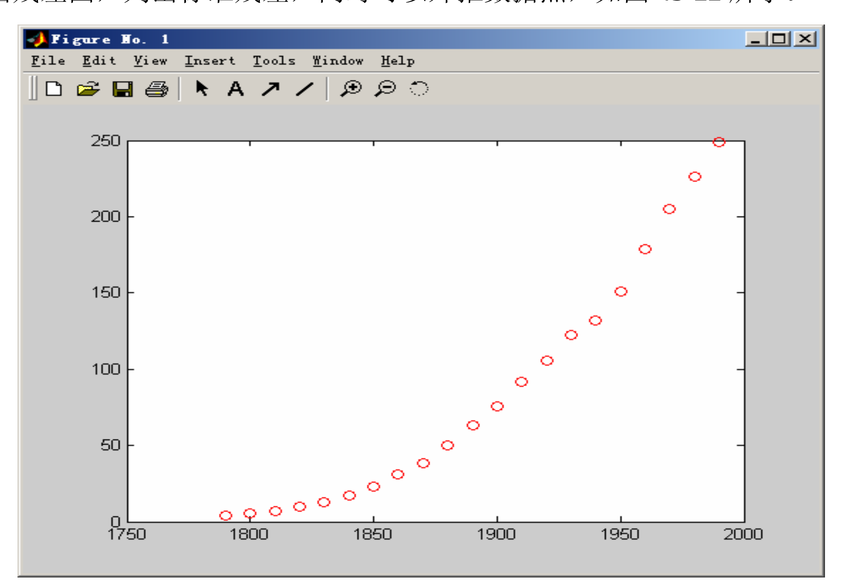


图 45-20　散点图

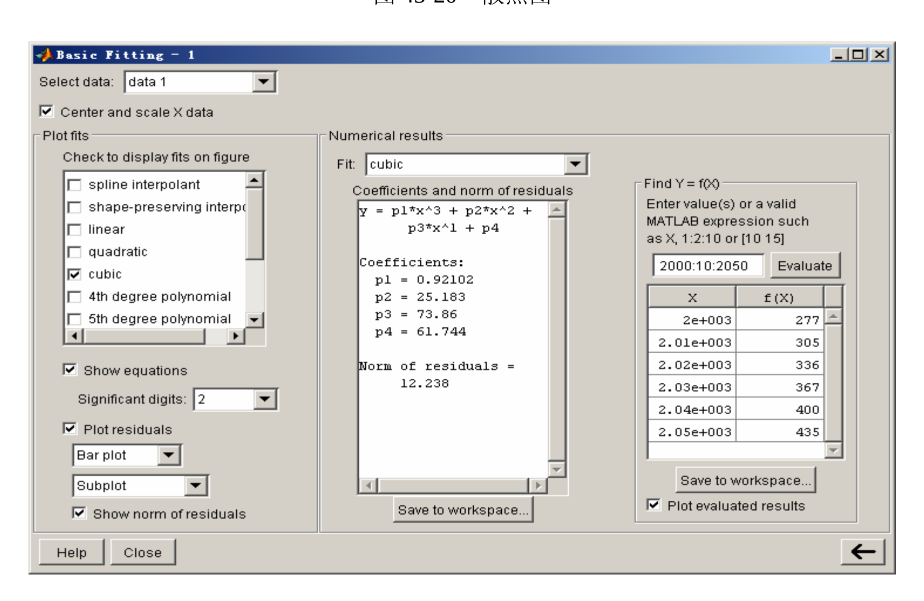


图 45-21　拟合参数的选择

图 45-22 显示了拟合图形、数据点、外推点以及拟合的方程。图形的下方显示了标准残差的分布图。图形显示拟合结果较好。

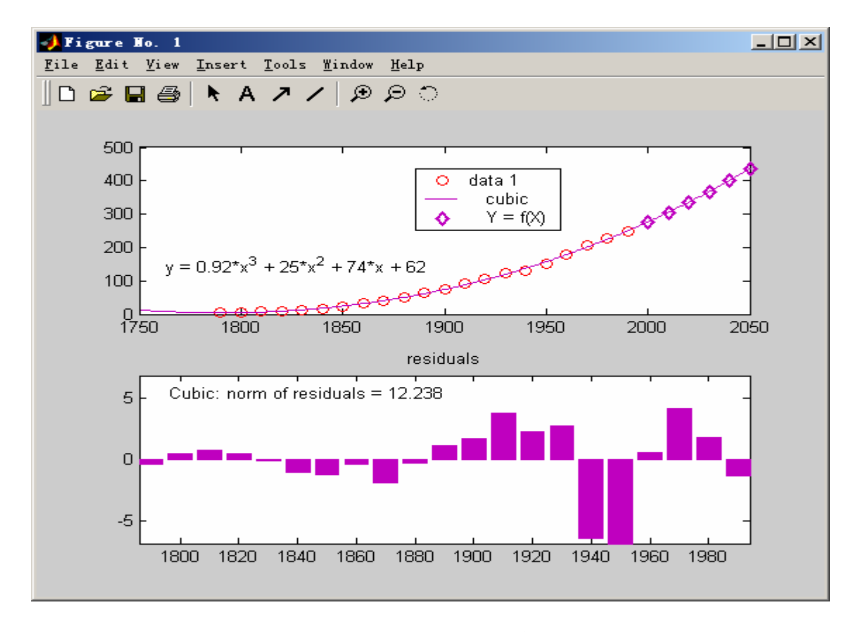


图 45-22　拟合曲线及残差表示图

参 考 文 献

（一）统计类

1 盛骤，谢式千，潘承毅编 . 概率论与数理统计 . 高等教育出版社，1994.4

2 苏金明，傅荣华，周建斌，张莲花编著 . 统计软件 SPSS 系列应用实战篇 . 电子工业出版社，2000.9

3 杜容骞编著 . 生物统计学 . 高等教育出版社

4 卢淑华编著 . 社会统计学 . 北京大学出版社，1989.8

5 史秉璋编著 . 医用多元分析 . 北京人民卫生出版社，1988

6 屠其璞编著 . 气象应用概率统计学 . 气象出版社，1984

7 [美] R.G.D. 斯蒂尔，J.H. 托里著，杨纪珂，孙长鸣译 . 数理统计的原理和方法（适用于生物科学）. 科学出版社，1979.5

8 卢崇飞，高惠璇，叶文虎编著 . 环境数理统计学应用及程序 . 高等教育出版社，1988.6

9 罗积玉，邢瑛编著 . 经济统计分析方法及预测 . 清华大学出版社，1985.7

10 张厚粲编著 . 心理与教育统计学 . 北京师范大学出版社，1988.6

11 董德元，杨节编著 . 实验研究的数理统计方法 . 中国计量出版社，1987.12

12 郑用熙编著 . 分析化学中的数理统计方法 . 科学出版社，1986.10

13 北京林学院编 . 数理统计 . 中国林业出版社，1982.3

14 华东水利学院编 . 水文学的概率统计分析 . 水利出版社，1980.8

15 周华章编著 . 工业技术应用数理统计学(上、下册) . 高等教育出版社，1965.3

16 邓勃编著 . 数理统计方法在分析测试中的应用 . 化学工业出版社，1984.11

17 曾秋成编著 . 技术数理统计方法 . 安徽科学技术出版社，1981.1

18 丁士晟编著 . 多元分析方法及其应用 . 吉林人民出版社，1981.12

19 王学仁，王松桂编著 . 实用多元统计分析 . 上海科学技术出版社，1990.9

20 R.V. 豪格，A.T. 克莱格著，朱宏道译 . 数理统计导论 . 高等教育出版社，1990.6

21 [美] H.L.阿尔德，E.B. 罗斯勒著，胡崇能译 . 概率与统计导论 . 北京大学出版社，1984.6

22 周纪芗编著 . 回归分析 . 华东师范大学出版社，1993.3

23 [美] 斯图尔特 L. 迈耶著，周鋆元，顾兆良译 . 科技工作中的数据分析 . 原子能出版社，1983.7

24 中国建筑工业出版社编 . 建筑工程质量标准 . 中国建筑工业出版社，1997.10

25 金沛焕编著 . 医用统计方法 . 上海医科大学出版社，1992

26 郝德元编著 . 教育与心理统计 . 教育科学出版社，1982

27 杨书勤编著 . 卫生统计学 . 北京人民卫生出版社，1993

28 马良驹，袁灿勤编著 . 岩土工程勘察数据统计分析 . 南京大学出版社，1991.1

29 沈毅，严曰树编著 . 医学统计学 . 上海医科大学出版社，1999.9

30 胡良平编著 . 现代统计学与 SAS 应用 . 军事医学科学出版社，2000,8

31 [加]Jiawei Han，Micheline Kamber 著，范明，孟小峰等译 . 数据挖掘概念与技术 . 机械工业出版社，2001.11

32 史忠植著 . 知识发现 . 清华大学出版社，2002.1

（二）优化类

1 郭科，胥泽银编著 ． 最优化方法及其程序设计 ． 四川科学技术出版社，1998.8
2 郭耀煌等编著 ． 运筹学原理与方法 ． 西南交通大学出版社，1994.9
3 席少霖等编著 ． 最优化计算方法 ． 上海科学技术出版社，1983
4 胡毓达编著 ． 非线性规划 ． 高等教育出版社，1990
5 薛履中编著 ． 工程最优化技术 ． 天津大学出版社，1989
6 刘惟信等编著 ． 机械最优设计 ． 清华大学出版社，1986
7 马国瑜编著 ． 化工最优化基础 ． 化学工业出版社，1982
8 张有为著 ． 动态规划 ． 湖南科学技术出版社，1991.11

（三）、样条类

1 卢传贤编著 ． 实用计算机图形学 ． 西南交通大学出版社，1996
2 唐泽圣编著 ． 计算机图形学基础 ． 清华大学出版社，1995
3 孙家广编著 ． 计算机图形学（新版） ． 清华大学出版社，1995

（四）偏微分方程数值解

1 梁昆淼编著 ． 数学物理方法(第三版) ． 高等教育出版社，1998.6
2 尚岳全，黄润秋著 ． 工程地质研究中的数值分析方法 ． 成都科技大学出版社，1991.8
3 冯肇华编著 ． 有限单元法基础 ． 吉林人民出版社，1984
4 李俊亭编著 ． 地下水数值模拟 ． 地质出版社，1992.10
5 孙纳正著 ． 地下水的数学模型与数值方法 ． 地质出版社，1982
6 程守洙，江之永等编 ． 普通物理学（1，2，3） ． 高等教育出版社，1993.8

［Ｇｅｎｅｒａｌ Ｉｎｆｏｒｍａｔｉｏｎ］
书名＝Ｍａｔｌａｂ工具箱应用
作者＝请自己补充
页数＝请自己补充
ＳＳ号＝请自己补充ＳＳＩＤ
出版日期＝请自己补充
出版社＝请自己补充